Micro Technology Unlimited

WORDPIC™

WORD PROCESSING SYSTEM

REFERENCE MANUAL

280010

May 1983

Rev. A

COPYRIGHT NOTICE

COPYRIGHT 1983, MICRO TECHNOLOGY UNLIMITED

This product is copyrighted. This includes the verbal description, programs and specifications. The customer may only make BACKUP copies of the software for his/her own use. The copyright notice must be added to and remain intact on all such backup copies. This product may not be reproduced for use with systems which are sold or rented.

Micro Technology Unlimited
2806 Hillsborough Street
P.O. Box 12106
Raleigh, NC 27605 USA
(919) 833-1458

DISCLAIMER OF ALL WARRANTIES AND LIABILITY

No warranties, either express or implied, are made by Micro Technology Unlimited with respect to this manual or the software described herein, its quality, performance, merchantability, or fitness for any particular application. This product is sold "as is". The buyer assumes all risk as to quality and performance. Under no circumstances will MTU be liable for direct, indirect, incidental or consequential damages resulting from any defect in the software, even if MTU has been advised of the possibility of such damages. Should the software prove defective following purchase, the buyer assumes the entire cost of all necessary servicing, repair or correction and any incidental, indirect, or consequential damages. Additional rights vary from state so some of the above exclusions and limitations may not apply to you.

NOTICE

Micro Technology Unlimited reserves the right to make changes to the product and specifications described in this manual at any time without notice.

TABLE OF CONTENTS

1.0 WORDPIC SYSTEM OVERVIEW - 1
1.1 Feature Summary - 1
1.2 Relationship of the Programs - 3
1.3 The CODOS Operating System - 3
1.4 The WORDPIC Shell - 4
1.5 The WPEDIT Editor - 5
1.6 The PAGINATE Page Formatter - 6
1.7 The PRINT Program - 7
1.8 Major System Utilities - 7

2.0 WORDPIC SHELL- 8
2.1 Starting the Shell - 8
2.2 General Operation - 9
2.3 EDIT Function Key - 10
2.4 NEWFILE Function Key - 11
2.5 PAGINATE Function Key - 11
2.6 PRINT Function Key - 12
2.7 COPY Function Key - 12
2.8 FILE ATT Function Key - 13
2.9 SEL DISK Function Key- 13
2.10 CHG DSK O Function Key - 13
2.11 NEW DISK Function Key - 14
2.12 DELETE Function Key - 15
2.13 BACKUP Function Key - 15
2.14 EXIT Function Key - 15

3.0 WPEDIT - 16
3.1 Starting WPEDIT - 16
3.2 General Operation - 18
3.3 Entering and Basic Editing of Text - - - - - - - - - - - - - - - - - - - 22
3.4 FIND Function Key - 24
3.5 CHANGE Function Key - 26
3.6 Code characters - 27
3.7 Code lines, General - 29
3.8 Tabstop Code line - 29
3.9 Paragraph Code line - 30
3.10 Newpage Code line - 30
3.11 Figure Code line - 30
3.12 Code line Commands and Attributes - - - - - - - - - - - - - - - - - - - 31
3.13 JUSTPARA Function Key - 40
3.14 Marking and Manipulating Text Blocks - - - - - - - - - - - - - - - - - 40
3.15 RECALL Function Key - 42
3.16 CENTER Function Key - 42
3.17 MACRO Function Key - 42
3.18 UPDATE Function Key - 43
3.19 QUIT Function Key - 44

4.0 PAGINATE - 45
4.1 Introduction - 45
4.2 Overview of the Pagination Process - - - - - - - - - - - - - - - - - - - 45
4.3 Definitions of Terms - 46
4.4 Startup Dialog - 47
4.5 The PAGINATE Display - 48
4.6 Headers, Footers, and Page Numbers - - - - - - - - - - - - - - - - - - - 50
4.7 SHO VAR and TEMPLATES Function key - - - - - - - - - - - - - - - - - - 54
4.8 BACK PG Function Key - 55
4.9 Alternate Facing Pages - 55
4.10 Paginating Multicolumn Text - 55
4.11 Notes for Successful Pagination - 56

5.0 NEC PC-8023 PRINT PROGRAM- 57
5.1 Overview of the NEC PC-8023 Printer- 57
5.2 Starting PRINT - 59
5.3 Program Action While Printing - 61
5.4 Return After Printing - 63
5.5 Interpretation of the Code Characters - 64
5.6 Interpretation of the Code lines - 65
5.7 Interpretation of the Code line Parameters - - - - - - - - - - - - - - - - - 67
5.8 Printing Headers and Footers - 72
5.9 Proportional Printing - 72
5.10 Tables and Charts - 74

6.0 MAJOR SYSTEM UTILITIES - 77
6.1 Running Utility Programs - 77
6.2 PRINT_NF = No Frills Print - 79
6.3 FILES = List All Files on Disk - 80
6.4 DIR = List Selected Files on Disk - 80
6.5 COPYF = Copy Single or Multiple Files - - - - - - - - - - - - - - - - - - - 82
6.6 DELETE = Erase a Specified File - 82
6.7 FORMAT+COPYF - Backup Onto Disk With Different Sidedness - - - - - - - - - 83
6.8 GGET = Get Graphic File to Screen - 83
6.9 GSAVE = Save Screen as a Graphics File - - - - - - - - - - - - - - - - - - 84
6.10 GETDIM = Display Dimensions of a Graphics File- - - - - - - - - - - - - - 85
6.11 GCONVERT = Convert Loadable File to Graphics File - - - - - - - - - - - - 85
6.12 ANNOTATE = Interactive Graphic Titling Program- - - - - - - - - - - - - - 86
6.13 GREDIT = Interactive Graphic Editor - - - - - - - - - - - - - - - - - - - 89

7.0 APPENDIX - 96
7.1 WORDPIC System Character Code Chart - 96
7.2 Graphic File Header Format - 97
7.3 Contents of WORDPIC Distribution Disk - - - - - - - - - - - - - - - - - - - 98
7.4 Graphic Density Chart - 99

8.0 INDEX - 100

PREFACE

This WORDPIC Reference Manual provides a complete operational and technical specification for the four major programs that make up the WORDPIC Word Processing system. In addition, it summarizes the use of several additional MTU-130 utility programs that are useful in word processing operations. In order to be complete, the language of this reference manual is necessarily more involved than that of the WORDPIC Tutorial Manual. Also the information is of a descriptive nature rather than a "how to" tutorial. Nevertheless, this manual should be usable by anyone with a general knowledge of typographic principles and a rudimentary knowledge of computer operations.

This manual should be used whenever you have a question about any aspect of WORDPIC operation that is not answered in the Tutorial Manual. Additionally, it provides details about system features that may be too infrequently used to be covered in detail in the Tutorial Manual. It is also useful in discovering the cause of unexpected system behavior or errors. The most useful section of this manual is Chapter 5 which discusses the Print program which has been customized specifically for the printer you ordered with your MTU-130. This is the only place where operations specific to your printer are discussed since printers vary considerably in their capabilities and limitations.

1. WORDPIC SYSTEM OVERVIEW

This chapter gives an overview of the WORDPIC system from a somewhat more technical viewpoint than the Tutorial Manual. Each element of the system is summarized here and then described in detail in subsequent chapters.

1.1 FEATURE SUMMARY

WORDPIC is an integrated system of word processing programs designed specifically for the MTU-130 computer. It allows the MTU-130 to be used as a professional word processor that can easily handle documents ranging in size from single mailing labels to reports exceeding 400 pages in size.

For speed, WORDPIC operates as an in-memory editor until the text exceeds available memory at which point it becomes an "infinite scrolling" disk file editor. Because of the high disk speeds of the MTU-130 and WORDPIC's interrupting keyboard feature, this is done quickly and transparently without interrupting operator entry. Any portion of the document can be rapidly retrieved to the display using automatic, bi-directional scrolling.

The display shows 80 character lines, with the ability to edit lines of up to 160 characters. The full ASCII character set is displayable, including lower case letters with descenders as well as the complete Greek alphabet, mathematical symbols, and a line drawing symbol set. The MTU-130 cursor controls are utilized in performing full screen editing. The text cursor can be moved rapidly with the five separate cursor keys to any position on the screen or in the file. When held down, these keys repeat automatically. A vertical and horizontal cursor key held down at the same time will perform diagonal movements.

Text entry is accomplished in either overstrike or insert mode. In both modes, as text entry on a line overflows the right margin, automatic word-wrap moves only whole words down to the next line. The keyboard is interrupt driven with N-key rollover and type-ahead buffering. This insures that no keystrokes are lost during word-wraps or disk accesses and permits error-free high speed typing by experienced operators. The INSERT and DELETE keys on the main keyboard, separate cursor cross, and function keys allow rapid editing for beginner and expert typists alike.

WORDPIC is one of very few word processors that allows text and graphics to be printed in one pass on the same document page. In fact its very name implies the conjunction of these fundamental human communication techniques. Figures up to 2048 dots wide by 2,048 dots high printed at densities up to 69,120 dots per square inch (usually limited by printer capabilities) may be handled by the system. In addition, there are two ways to handle mathematical equations.

WORDPIC shows an alphabetized list of all text and graphic files when the program begins. Files are selected from this index by moving a cursor to the desired file and pressing the appropriate function key. Because the function key legends are always visible at the bottom of the display, no memorization of "control keys" or complex key sequences is necessary. This makes learning the system easy for beginners while maximizing the speed of experts.

You can specify formatting commands for PITCH, SPACING, JUSTIFICATION, MARGINS, TABS, PAGE LENGTH, HEADERS/FOOTERS, PAGE NUMBERING, and more. All parameters can be changed at any time, even for single lines. You can select Pica, Elite, condensed, or proportional type styles; ragged, flush right, or proportional justification; and line and paragraph spacing can be set in half-line increments.

Blocks of lines can be deleted, moved, saved, and recalled anywhere in the document. Also, blocks of text can be written to other files, and other files can be inserted anywhere.

Both tabstops and margins are easily set or changed as often as needed within the document, with the current settings shown graphically at all times. Both left and right margins can be reduced or expanded, with one key operation to automatically "funnel" or "expand" the text as needed to fit between the altered margins. "Reverse indented" paragraphs can be created and edited, making outline preparation a snap.

WORDPIC supports a powerful and flexible method for automatically printing headers and footers on a page. Automatic page numbering is also provided. Headers and footers do not appear in the document file, but are instead stored in a separate file called a "template" file. Only one copy of each header or footer is necessary in the template file. Subsequent changing of the headers and footers on all the printed pages in a document can be done by editing the template file and reprinting the document.

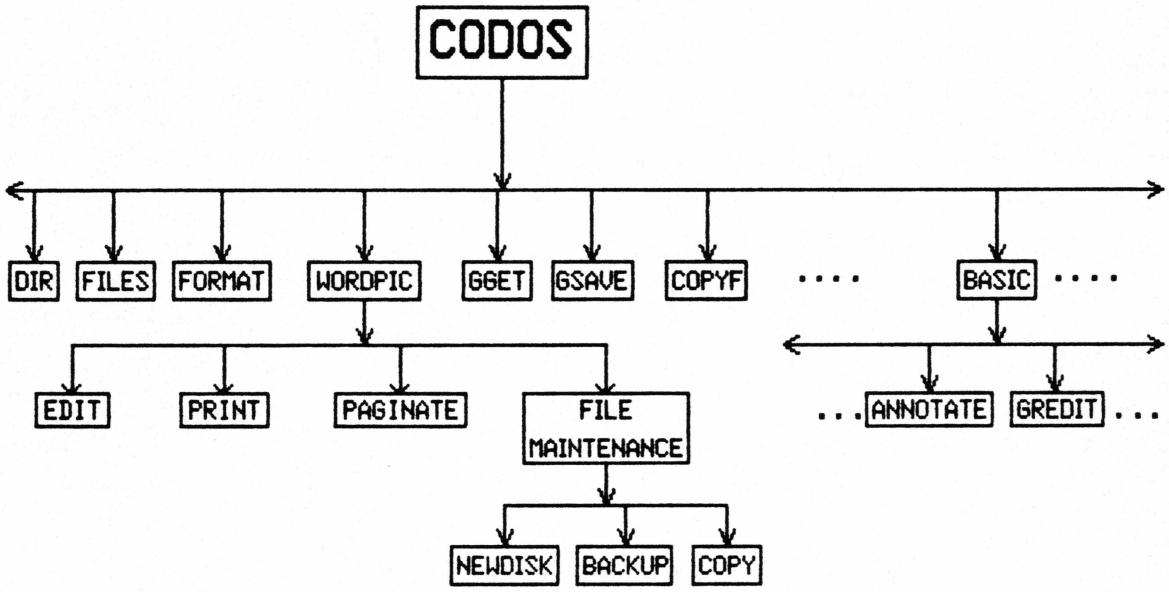
WORDPIC can perform full search and replace operations which the user may limit to specified columns or line ranges. Operator "veto" may be selected to allow approval of each occurrence of a string match prior to replacement. The search normally is case insensitive, but may be made case sensitive when desired.

A special feature allows you to "teach" WORDPIC new editing operations with up to 8 operations being defined and operational at any given time. These "macros" are automatically saved on disk so that they will be available in subsequent editing sessions using that same disk. Thus, you can customize WORDPIC to your own needs.

Combining the graphics and equation capability with WORDPIC's proportional spacing and printing gives documents a typeset appearance. WORDPIC presently supports the NEC 8023A printer for low cost flexibility, the Brother HR-1 for high quality monospaced text, Daisywriter for high quality proportional text, and the Santec S-700 for the utmost in flexibility with letter quality print. The source code for WORDPIC's companion print program is supplied on the distribution diskette. This allows additional printer support to be added by users in addition to MTU's ongoing developments.

1.2 RELATIONSHIP OF THE PROGRAMS

The WORDPIC software system for the MTU-130 computer is actually a collection of several programs. These individual programs (or modules) are well integrated so that they appear to the user as a single program, namely WORDPIC. The drawing below identifies the major components of the MTU-130 WORDPIC software system and shows how they relate to one-another.



1.3 THE CODOS OPERATING SYSTEM

At the top of the chart is the CODOS operating system. CODOS is an acronym for Channel-Oriented Disk Operating System. It is the primary program under which all MTU-130 programs run and are subservient to, whether word processing oriented or for other applications. The primary purpose of CODOS is to control the MTU-130 hardware, especially the disk drives, but also the display, the keyboard, the light pen, and even the sound generator. In controlling the disk drives, it keeps a catalog of data and program files on the disk and controls the allocation of space as files are created, deleted, and change in size. It is this file management function of CODOS that is most used by WORDPIC.

CODOS is the master gateway into the entire world of MTU-130 applications. WORDPIC essentially "hides" CODOS from the word processing user's view and thus makes the MTU-130 look like a dedicated word processor. It is possible to use CODOS directly however by using a CODOS disk rather than a WORDPIC disk or by exiting WORDPIC which returns you to CODOS's command processor. CODOS is fully explained in the MTU-130 Users Manual and really is easy to use compared to other small computer operating systems.

Since CODOS is used for file management, its rules regarding the naming of all disk files also apply to WORDPIC files. All file names have three parts: the name itself, the extension, and the disk drive number. The name is always required to be specified while the extension and the drive numbers need not be specified if the defaults that are assumed are acceptable.

The name must be at least 2 characters long and may be as long as 12 characters. It is made up of any combination of upper case letters A-Z, digits 1-9, or the underline (_) character which is useful for a separator since blanks are not allowed in a name. One restriction is that the first character of a name must be a letter.

The extension, if present, consists of a period followed by a single letter or digit. It is used as a reminder to the operator of what kind of data is in the file. When using WORDPIC, you normally don't need to specify an extension since all of the WORDPIC programs will assume an extension of .W for WORDPIC text files. For this reason, the .W doesn't even show up when WORDPIC displays a WORDPIC text file name. .G is the extension that normally denotes a file that contains graphic information. When dealing with graphic files, the .G must be specified because the default, .W, is not suitable. When running non-WORDPIC programs from CODOS, the file extension defaults are different such as .B for BASIC program files and .C for most other program files.

The disk drive number, if present, consists of a colon (:) followed by a single digit. For most systems with only two disk drives, the drive number may be only 0 or 1 but 3 and 4 drive systems may use numbers up to 3. The drive number simply identifies which drive holds the disk that has or will have the specified file. Automatic searching for a name among multiple drives is not done for security and safety reasons. Drive numbers are not normally needed when running WORDPIC because the currently selected drive is used as a default. When using utility programs under CODOS however, a drive number will be needed to access files on drives other than zero.

Below are some examples of valid file names. Of course you will want to use names that are descriptive and easy to remember or recognize for your own files.

T1 TEXTFILE1 TEXT_FILE_12 MAILLABEL.M RESUME_BAK:1 ARCHIVE342.T:2

The CODOS section of the MTU-130 Users Manual has a more detailed discussion of file names.

1.4 THE WORDPIC SHELL

Under CODOS there can be any number of programs or groups of programs. One such program group is the WORDPIC Word Processing System. Heading this group is the program called WORDPIC which serves to tie together the other word processing programs just like CODOS ties together the whole world of MTU-130 programs. A program that oversees such a group of related programs and implements its own operator interface independent of the computer's operating system is often called a shell program. This is because it effectively creates a shell that covers the harsh world of the operating system and related programs and thus "hides" it from the user. Although the proper full name is "WORDPIC Shell", it will usually be shortened to just WORDPIC in this manual.

Perhaps the main function of WORDPIC, besides calling the various word processing programs into action when needed, is producing a convenient, alphabetized display of all of the WORDPIC files (.W extension) and graphic files (.G) extension on the user's disk. This prevents clutter from other non word processing files that may be on the disk. The display format allows up to 100 files to be displayed at once, with provisions for "paging" through the list if necessary to show more than 100. A special file selection cursor can be moved around on the file display to quickly and easily select a particular file for editing, printing, or other processing. Except when creating a new file, seldom does the user have to type in a file name when using the WORDPIC Shell.

WORDPIC also provides several file maintenance functions which allow the user to copy files from one disk to another, delete files, prepare new disks right from the box for word processing use, and make total backup copies of disks for archiving. Each of these functions is performed by pressing function keys and, depending on the function, answering questions on the display. Additional file maintenance functions are available under CODOS and are described in Chapter 6.

As mentioned earlier, the WORDPIC Shell serves to insulate the user from CODOS. To do this effectively, it tries to anticipate every possible error condition prior to having CODOS open a file, create a file, etc. When such errors are detected, WORDPIC issues a comprehensive, easily understood error message to the operator and insures that the problem is corrected before continuing. Under any normal circumstances a WORDPIC user should never see a CODOS error message. Under some conditions, such as a disk becoming full or developing a bad spot, a CODOS error may be signalled. Its format is easily identified (a sample is shown below) and the message text is generally clear although terse.

```
CODOS ERROR 06
P=21B9:0/0 (001300) A=00 X=05 Y=EB F=21 S=F3
DRIVE NEEDED IS NOT READY.
```

Of course the numbers and the message will be different. If the error is due to a "bug" in one of the word processing programs, the numbers on the second line will be particularly important.

Finally, the WORDPIC Shell provides a convenient exit to CODOS via a function key. WORDPIC may be restarted by simply typing WORDPIC in response to a CODOS> prompt. Normally the WORDPIC command is part of the special "startup" file on WORDPIC disks so that WORDPIC will automatically be run by CODOS whenever the MTU-130 is turned on or cold reset, provided that a WORDPIC disk is in drive 0 at the time.

1.5 THE WPEDIT EDITOR

The editor for the WORDPIC system is called WPEDIT. It is actually a separate program which is usually called into action by the WORDPIC Shell but can also be run independently from CODOS. Much of the time spent with the WORDPCI system will be spent with the WPEDIT program.

The purpose of WPEDIT to create or edit a word processing text file interactively under direct operator control. This is how all original word processing text is entered into the system and how virtually all revisions (except pagination) are done. WPEDIT works by editing a file "in-place" which means that the original file itself is being changed directly; no temporary copies or work files are created in the process. If desired, however, WPEDIT can make a copy of the original file before editing it. If made, this is called a "backup copy" and may be valuable in the event of an editing "accident" or a machine failure.

WPEDIT is screen oriented and function key driven. This means that operation is largely intuitive and the results of changes can be seen immediately and in the proper context of surrounding text lines. Lines up to 80 characters wide are handled in normal mode while up to 160 character lines may be handled in "landscape" mode. The text file is presented on the screen as a "scroll" in which 20 lines are visible at any one time and the invisible portion is "rolled" up at the top and bottom edges of the visible screen. The size of the scrolled file changes dynamically as text is added and deleted. The upper size limit approaches 1 million characters (about 150,000 words or several hundred printed pages) while the lower size limit is 1 character (if all of a file's text is deleted, the file will disappear when exiting WPEDIT).

While entering text, words that are too large to fit at the end of a line are automatically "wrapped" to the next line thus avoiding the broken appearance seen in some other editors. Word wrap is even active during insertions. After heavy editing, such as multi-character insertions or deletions or changing the margin settings, the current paragraph may be "closed up" or re-justified right on the screen by pressing a single function key. While this results in a ragged looking right margin, the PRINT program is capable of producing a flush right margin by adjusting word and character spacing if desired.

WPEDIT can also read and write files other than the one you are currently working on. You can "recall" any other text file on the same or a different disk and have its contents inserted at any point in the file being edited. Thus the merging of stock paragraphs with original text in a document is easily accomplished. Also, any portion of the current work file may be designated and then saved as a separate file to be used later. An important feature is that these external files need not be WORDPIC text files but could instead be ordinary text files, the output of an analysis or computation program, or even the input to some other program. In this regard WORDPIC and WPEDIT meshes well with the entire range of possible MTU-130 applications.

In order to meet the special needs of high speed entry of text, WPEDIT has its own keyboard driver rather than use the "standard" keyboard driver offered by CODOS. This special driver implements N-key rollover which means that a new keystroke may be recognized even though the previous key or keys are not yet released. It also makes the keyboard operate on an interrupt basis which means that keystrokes will not be lost even though the computer may be "busy" performing a word-wrap or accessing the disk. Interrupt operation is superior in many respects to the so-called "buffered" or "intelligent" keyboards used on the newer advanced computers and is much superior to the polled keyboards common on older systems.

Whenever WPEDIT is run, it uses a file called WORDPICSAVES.Z on the same disk as the file being edited. This file is used to automatically save deleted text for possible recall later in case the deletion was a mistake. Its most valuable use however, is to hold the 8 recall buffers and 8 macro definitions that may be created. These buffers and definitions then persist from one WPEDIT session to the next even if the system is powered off. Thus macros and saved phrases may be set up once and then used whenever any file on that disk is edited.

1.6 THE PAGINATE PAGE FORMATTER

For printing on standard cut sheet paper, the long text scroll produced by WPEDIT must be cut into pages of appropriate length. This may in fact be accomplished in WPEDIT by manually inserting "page breaks" where desired. PAGINATE is a program which can do this semi-automatically with the operator making the "judgement calls" necessary to give a document a professional, finished appearance.

PAGINATE works interactively by computing pagebreak locations according to certain rules and then allowing the operator to accept or modify these locations one at a time. Several lines preceeding and following proposed pagebreaks are displayed on the screen thus showing the context of the pagebreak. If the operator elects to alter the proposed break, the display continuously shows how many lines over or under the desired number of lines per page the altered break will be if accepted. PAGINATE can also remove all existing pagebreaks in a document for complete repagination or re-propose those that already exist. Note that a file that has been processed by PAGINATE may still be edited by WPEDIT just like an unpaginated file. The only difference is that a large insertion may cause that particular page to overflow when printed.

What makes PAGINATE particularly valuable is that it takes into account all of the variables that affect the placement of text on the page when computing a proposed page break location. Thus the final printed height of figures is considered, the effect of changing line spacing, the effect of Backup and Backpage commands embedded in the text and other parameters are all accounted for.

1.7 THE PRINT PROGRAM

The PRINT program actually prints a word processing file on paper. It is far more sophisticated than an ordinary file lister since it must interpret all of the various codes and commands that may be embedded in the text. In the case of figures, it must locate the designated graphic file and print the bit image in the specified density. After interpreting the codes, commands, and graphic data, it must translate them into a sequence of printer actions to accomplish the desired effect. Since printers vary widely in their capabilities and command sets, there is a different PRINT program for each kind of printer WORDPIC supports. The use of custom PRINTs is an important feature because it allows text files to be largely printer independent even when "special effects" are being used.

Another function of PRINT is to add headers, footers, and page numbers to each page when desired. The headers and footers come from a separate file and therefore may be changed without editing the actual document text. Headers and footers may be one or several lines long and can even include a graphic figure, such as a company logo. Page numbers may be printed wherever desired and may even be "compound" such as 4-23.

Final justification of text margins is also performed by PRINT. The JUST PARA function of WPEDIT performs the usual ragged right justification by putting as many words as possible on each line. PRINT then prints the lines either as is for a ragged right appearance, with "flush justification" where interword spacing is adjusted to achieve an even right margin, or with "proportional" justification where both word and character spacing is adjusted. The latter option, when performed on a letter quality printer having a proportionally spaced font, results in a typeset appearance. Unlike other systems with proportional spacing, WORDPIC PRINT can justify in "zones" so that reverse indented material and numbered lists will line up on the left as well as the right. PRINT does not move material from one line to the next when justifying like some systems so the final printed format is always predictable.

1.8 SYSTEM UTILITY PROGRAMS

Besides the major word processing programs that may be accessed from the WORDPIC Shell, there are a number of utility programs that are useful in word processing operations that may be accessed only from CODOS. Only the most immediately useful utilities will be described in this manual in Chapter 6. These include extended file maintenance utilities (FILES, DIR, FORMAT, COPYF, and BACKUP), utilities for manipulating graphic files (GGET, GSAVE, GETDIM, and GCONVERT), and graphic image creation programs (GREDIT and ANNOTATE). Many others are described in the MTU-130 Users Manual. In addition, new utility programs are constantly being released on MTU-130 User's Group disks.

2. WORDPIC SHELL

The WORDPIC Shell is a program called WORDPIC which serves to integrate the other word processing programs together and insulate the user from the CODOS operating system. WORDPIC is usually started automatically by "booting" a WORDPIC word processing diskette but it can also be started by entering a command into CODOS.

2.1 STARTING THE SHELL

The WORDPIC Distribution disk supplied by MTU has its STARTUP.J file set up so that whenever the computer is turned on or cold reset with a copy of that disk in drive 0, WORDPIC will start automatically. This has been accomplished by placing the command: WORDPIC in the last line of the STARTUP.J file. Below is what the screen will look like when WORDPIC starts automatically, before the date is entered:

```
                    tm
MTU WORDPIC   WORD PROCESSING SYSTEM

(C) Copyright 1983, M.T.U.

Printer support is for NEC 8023 printer


WORDPIC Shell program will start automatically once date is entered.

CODOS 2.0   (C) 1981 MTU
PLEASE ENTER DATE (EXAMPLE:04-JUL-76)?=
```

WORDPIC may also be manually started from CODOS by simply entering: WORDPIC and a carriage return in response to a CODOS> prompt. This will initiate normal operation and allow a convenient exit back to CODOS. If desired, exit from WORDPIC back to CODOS can be suppressed by following the WORDPIC command with a blank and the letters NE for No Exit.

When WORDPIC begins, it closes disk drive 1 and then it checks the disk in drive 0 for the existence of certain files to prevent possible errors later. These required files are listed below:

WORDPIC.C	NEWDISK.C
WPEDIT.C	WSBACKUP.C
PRINT.C	WORDPICSAVES.Z
PAGINATE.C	The CODOS system overlays

If any of these files is missing, a message to that effect is displayed and you will be asked to insert a different disk into drive 0 that has all of the required files.

Once WORDPIC is started and the necessary files are verified to exist, the index of drive 0 will be searched and all of the files with .W or .G extensions will be extracted, alphabetized, and displayed on the WORDPIC index display. An example of a typical display is shown below:

```
                 WORDPIC Shell 1.0    Copyright 1983 M.T.U.
Please select a file with cursor keys, then press a function key.
 A_BLANK_MEMO      MEMO_03_10
 BOOK_CHPT_1       MEMO_03_19
 BOOK_CHPT_2       MEMO_03_30
 BOOK_CHPT_3       MEMO_04_17
 BOOK_CHPT_4       MEMO_04_27
 DEC_82_ART        PRINT_DOC
 EXAMPLE_1         PRINT_DOC_BK
 EXAMPLE_2         PRINTFIGS
 FEB_83_ART        PRT_FIGURE_1
 FIGURE_1.G        SMITH_LTR_1
 FIGURE_2.G        SMITH_LTR_2
 FIGURE_3.G        WPEDIT1.G
 HINTS_DOC         WPINDEX.G
 JAN_83_ART        WPJUSTP1.G
 JONES_LETTER      WPJUSTP2.G
 MEMO_01_04        WPJUSTP3.G
 MEMO_01_13
 MEMO_01_21
 MEMO_02_08
 MEMO_02_28
  28 Text files,   8 Graphic files.   21% space left on disk in drive 0.
 | EDIT | NEWFILE | PRINT |  |   | PAGINATE | COPY | FILE ATT | OTHER |
```

The top line of the display is the program ID and copyright notice line. This is true in all of the WORDPIC system programs so that the operator can quickly identify which program is running. If an error condition occurs, this line is used temporarily to show the error message. If the operator is to enter information, such as a file name, the entry is automatically shown on this line.

The second line is the operator prompt line. It is typically used to instruct the operator to do something such as select a file, press a function key, or enter some information. All prompts are self-explanatory and phrased in proper English.

The index display area is surrounded by a box and contains as many as 5 columns of 20 file names each. Only the names of WORDPIC text files (.W extension) and graphic files (.G extension) are shown and these are alphabetized as a group. The graphic files can be distinguished because the .G portion of their name shows while the text files do not show an extension. Alphabetizing is left-to-right with underline coming first, then the digits, and finally the letters. If there is both a .W a .G file of the same name, the .G file is listed second. The index display area will hold up to 100 file names. If more than that are present, the legend for function key f4 will change from blank to INDEX to allow paging through the index display. Over 200 files can be stored on a disk if they are fairly short (average of one single-spaced page each).

Below the index display is the disk summary line which shows the total number of text files and the total number of graphic files on the currently selected diskette as well as which disk is currently selected. It also shows how much disk space remains for new files or expansion of old files in percent. A disk with less than 5% free space should be split into two disks before any significant insertions or new files are added. At approximately 2% free space, the WORDPIC system programs will warn of the critically limited free space.

Most WORDPIC Shell operations require that the operator first select one of the files shown in the index display. The currently selected file is always highlighted with a reverse video "file cursor". This cursor may be moved around the screen with the cursor control (arrow) keys. Although the file cursor may be positioned to blank space, operations requiring a file name will display an error message if the cursor is not actually covering a file.

All WORDPIC operations are initiated by pressing function keys. At the bottom of the display are the current legends for the 8 function keys arranged in two groups of 4 each. There are two primary "menus" of function key legends called the "main menu" and the "other menu". The rightmost key (f8) is used to select between the two. Some of the function keys will display still other sets of function key legends which in all cases have descriptive titles.

After a particular WORDPIC function is selected with a function key and its own menu is displayed, f8 will always be labelled CANCEL. Pressing the CANCEL key will always terminate the function being performed and return to one of the two primary menus.

In order to prevent undesired error returns to CODOS, WORDPIC does extensive error checking during every step of each operation. Whenever one of these error conditions occurs, WORDPIC will display a detailed error message identifying the particular problem and instructing the operator in the corrective procedure. In most cases the operator will be instructed to correct the problem and then press the carriage return key to inform WORDPIC that corrective action has been taken. Some messages are only informative in nature and do not require any corrective action but RETURN must still be pressed to acknowledge the message. Note that pressing the CANCEL key (f8) is an alternative to immediately correcting the problem and can be used to cancel the operation that detected the problem instead.

2.3 EDIT FUNCTION KEY

In order to edit an existing file, the file cursor should be moved to cover the desired file name and then the EDIT function key pressed. The MAINMENU key (f8) may be pressed if EDIT is not seen. When EDIT is pressed, WORDPIC will ask whether you would like to keep a copy of the original, unedited text which is called a backup copy. The function key legends will show YES, NO, and CANCEL. CANCEL will terminate the edit function and immediately return to the main function key menu. NO will immediately start the WPEDIT program which allows editing the chosen file without leaving an unedited copy of it behind.

YES will ask for a new file name for the copy. If the name entered is the same as an existing file (perhaps a previous backup), WORDPIC will then ask if that file should be overwritten (replaced) by the backup. A NO response at that point will allow a different file name entry. The name must conform to the rules for file names (see section 1.3). Typically an extension will not be entered in which case WORDPIC will internally supply a .W extension. Note that if an extension is appended to the name and it is not .W or .G that the backup file will still be generated but it will not be displayed on the WORDPIC index display. To see a listing of such files, an exit to CODOS must be taken and the the FILES command used (see section 6.3).

Backup files are advisable until familiarity with the system is achieved and are always advisable when long or valuable documents are being edited. An accident such as a power failure while editing or the appearance of a bad spot on the disk (perhaps due to a fingerprint) could cause the original file to become somewhat fragmented. If such an event happens and a backup had been made, the most that can be lost are the effects of the current edit session. Following the backup dialog, WPEDIT, which is described in chapter 3, is entered.

2.4 NEWFILE FUNCTION KEY

In order to create a new word processing text file, the NEWFILE function key should be pressed. The position of the file cursor in this case is not significant. WORDPIC will then ask for a file name for the new file. The name entered must be unique, that is, it must not be the same as a name already on the currently selected disk. If a duplicate name is entered, an error message will be displayed and a different name entry requested. The CANCEL key may be used if you decide not to create a new file after all.

The name must conform to the rules for file names (see section 1.3). Since a primary word processing file is being created (as opposed to a backup copy), only a .W extension (or none at all in which case the .W will be appended by WORDPIC) is allowed. WORDPIC will also internally append the number of the currently selected disk drive to the file name. Since the other disk drives are closed while running WORDPIC, specifying a different drive as part of the file name should not be attempted.

After an acceptable file name is entered, WPEDIT will then ask whether the file should be handled in normal or wide mode. Normal mode allows text lines up to 80 characters long to be entered and edited while wide mode allows up to 160 character lines. The advantage of normal mode is that each text line is shown as one screen line and thus 20 lines may be viewed at once. Wide mode shows each text line as two screen lines which is somewhat more difficult to read and reduces the screen capacity to 10 text lines. Note that the whole file must be either one mode or the other. If normal is selected, it may later be changed to wide but if wide is selected, changing to narrow later will truncate any lines in the file that are wider than 80 characters.

After the width selection is made, the WPEDIT program is entered which will show a blank page. Note that if WPEDIT is immediately exited with the blank page still showing, i.e., no text at all is entered into the file, the file will be deleted and will not appear in the index display.

2.5 PAGINATE FUNCTION KEY

To automatically paginate the text in an existing file, the PAGINATE function key should be pressed. When PAGINATE is pressed, WORDPIC will ask whether a copy of the original, unpaginated (or previously paginated) file should be kept as a backup copy. This question is asked because PAGINATE, like WPEDIT, processes the text file in place without making any copies or work files itself in the process. See section 2.3 for a discussion of backup copies and a description of the dialog.

Next, the number of text lines per page will be requested. This establishes the amount of space on each page that will be reserved for file text at the rate of 1/6 inch per line exclusive of top and bottom margins and any headers or footers that may be called for. (Note: the number of lines specified is simply divided by 6 to establish a distance in inches; the actual number of text lines that will fit into that distance will depend on the SPACING and LPI parameters established in the file.) A typical number of lines is 54 which sets a text length of 9 inches thus allowing two inches for the top margin, header and footer (if any), and bottom margin. See section 4.3 and 4.4 for additional discussion.

The next question asks if all existing pagebreaks should be discarded before paginating. If a complete pagination is being performed, Yes is used which will delete all existing pagebreaks in the file. If only minor changes to existing pagination is contemplated, No is used which will re-propose each existing pagebreak. After this question is answered, PAGINATE will be entered. See section 4.

2.6 PRINT FUNCTION KEY

To print the text of an existing file, the PRINT function key should be pressed. When PRINT is pressed, WORDPIC will ask how much of the file should be printed and display several function key legends for options. Some of the options will then ask for "page numbers" to be entered. Note that these refer to a sequential page count determined by counting Newpage code lines in the document and may not correspond to page numbers that may be printed as part of a header or footer. The choices are ONE PAGE (will print only the specified page), FROM PAGE (will print from the specified page to the end), RANGE (will start and stop at designated pages), and ALL. To print from the beginning to a specified page, use the RANGE option and specify page 1 as the first page.

Next WORDPIC will ask whether the printer should use "rough draft" or "final" mode for printing the document. The exact difference (if any) between these two choices is highly printer dependent and Chapter 5 should be consulted for details applicable to your particular printer. In general though, rough draft will print faster and not as dark or clear as final and will cause less wear on the ribbon and printer mechanism.

At this point PRINT is entered which will ask an additional question before printing actually begins. See section 5.2 for the discussion. After PRINT completes its task, it will return to WORDPIC. If printing was aborted prematurely or an error is detected, the return to WORDPIC will be accompanied by an error message. Carriage return must then be pressed to acknowledge this message before continuing with WORDPIC.

For operational simplicity, the WORDPIC system does not support simultaneous printing and editing of another document. Some printers, however, have a large internal buffer that may hold up to several pages and thus will "finish" from PRINT's point of view long before the physical printing is actually complete.

2.7 COPY FUNCTION KEY

The COPY function key can be used to make a copy of a file to the same or a different disk using the same or a different name. Before COPY is pressed, the name of the file to be copied should be selected with the file cursor. When COPY is pressed, the first question asks which disk drive to write the copy to (the destination drive). The possible answers are displayed on the function key legends and one of these (or the "0" or "1" key) should be pressed.

If the destination disk is different from the currently selected disk (source disk), WORDPIC will ask if the same file name should be used. If not, a new file name for the copy will be requested. If the source and destination disks are the same, a new file name will always be requested since all names on a disk must be unique. The file name may have an extension other than .W but may not have a drive number as part of the name (see section 2.3).

After the file name is entered, it is checked for uniqueness, the operator is requested to insert a disk into the destination drive if necessary, and the file is copied. The COPY function key does not change which drive is currently selected. Only one file at a time may be copied using the WORDPIC copy function. For batch copying of the entire contents of a disk, see the description of the BACKUP function key in section 2.12. For batch copying of just files with related names, see the description of the COPYF utility in section 6.6.

2.8 FILE ATT FUNCTION KEY

The FILE ATT function key will display and allow alteration of certain word processing file attributes. Before pressing FILE ATT, the desired file name should be selected with the file cursor. After FILE ATT is pressed, the various file attribute functions available are then displayed. Each of these is described below.

DATE? This displays the creation date (not the last edit date) of the selected file.

SIZE? This displays the percentage of total disk space occupied by the file. On single-sided disks, 1% is approximately 5K bytes. On double sided disks 1% is approximately 10K. Files are allocated space in blocks of 0.4% each.

LN WDTH? This displays the maximum line length allowed by the file as determined when the file was created. The length may not be 80 or 160 if the file was created by calling WPEDIT from CODOS rather than from WORDPIC.

CHG DATE This allows the creation date of the file to be changed without having to copy the file and delete the original. Although any 9 character string is accepted for the date, the format DD-MMM-YY is preferred.

CHG WDTH This allows the maximum line length of the file to be changed. A warning is given if the line length is decreased. See section 2.4 for a discussion of line length.

RENAME This allows the name of the selected file to be changed. The usual rules for file names apply. Any extension is accepted but if it is not .W or .G or blank, the new name will not be shown on the index display. See section 1.3 and 2.3 for details about file names. After the file is renamed, the main menu display is restored.

2.9 SEL DSKX FUNCTION KEY

The SEL DSKX function key (where "X" is either 0 or 1) allows a different drive to be selected as the "current" drive and its index displayed. The number displayed as part of the legend is always opposite that of the currently selected drive. When SEL DISX is pressed, the operator is requested to insert a disk into the newly selected drive, close the door, and press RETURN. The disk index is then read and the file names with a .W or .G extension are displayed. Note that when disk 0 is selected that diskettes may be freely inserted and removed from drive 1 but when disk 1 is selected, drive 0 still must not be disturbed since the WORDPIC system files are on that disk. The CHG DSK 0 function key should be used for changing the disk in drive 0.

2.10 CHG DSK 0 FUNCTION KEY

The CHG DSK 0 function key allows the diskette currently in drive 0 to be removed and replaced with a different diskette. There must always be a diskette in drive 0 and it must have all of the WORDPIC system files on it when WORDPIC is being used. See section 2.1 for a list of files required to be on any disk in drive 0 while WORDPIC is running. The CHG DSK 0 function will always make the disk in drive 0 the current disk. When the index is redisplayed, it will show the files on the new disk drive 0.

2.11 NEW DISK FUNCTION KEY

The NEW DISK function key will erase a diskette in drive 1 and then prepare it for use by the WORDPIC system. New diskettes fresh from a box must be erased before they can be used on the MTU-130 (the BACKUP function key and the BACKUP and FORMAT utilities can also be used to erase a disk). NEW DISK can prepare either single-sided or double-sided disks for use regardless of the "sidedness" of the disk in drive 0.

When NEW DISK is pressed, WORDPIC first asks the operator to insert the disk to be erased in drive 1 and then press RETURN. The CANCEL key (f8) may be pressed instead at any time during the ensuing dialog to cancel the NEW DISK function. The disk is not actually erased or even touched until an affirmative response to the last question in the 5 question sequence is received.

WORDPIC first asks for a volume number to be written on the disk after it is erased ("volume" is a synonym for "diskette" in computer terminology). Any 4 digit number will do since volume numbers are not currently used for any purpose by WORDPIC or CODOS (although CODOS's "DISK" command can be used to display the volume number). Operators may wish to establish a volume numbering system of their own however.

The next question asks whether the new disk should be checked for defects after it is erased. Although virtually all name brand diskettes are advertised as being "100% error free", it is advisable to check for defects until experience with a particular brand is established. If bad spots are found, they are reported and then "bypassed" which means that they will not be used for storing data and a slight loss in disk capacity will result. If there are damaged areas on the disk and checking is not done, part of a text file may later become unreadable and thus partially or completely lost. The only advantage of answering NO to this question is that 8 minutes (4 minutes for single-sided disks) of time is saved in erasing. The critical index and boot track areas of the diskette are always checked regardless of how this question is answered.

The next question asks whether the new disk will be used in drive 0 after it is erased. If the answer is YES, the essential WORDPIC system files will be copied to it after it is erased. These files will occupy 28% of the capacity of a single-sided disk or 16% of a double-sided disk. If the answer is NO, the disk may only be used in drive 1 but nearly 100% of its space will be available for user files.

The next question asks if large files will ever be recorded on the new disk. This question may be answered either way with no ill effects and no change in the maximum document size that can be recorded on the disk. A NO answer will maximize the speed at which files up to about 10% of the disk capacity in size can be manipulated by WPEDIT. Files larger than about 10% will however experience substantial slowdowns (a factor of nearly 5) near the file's end on disks prepared for small files. A YES answer will slightly slow the manipulation of all files (by about 30%) but that speed will be maintained for any size file, even one that approaches 100% of the disk capacity.

The final question asks if the user truly desires to erase the disk in drive 1. If answered "YES", the erase sequence begins. Roughly 1 minute will be required if defect checking is not performed. If any bad spots in the index areas of the disk are detected, the disk is declared to be unusable and should be discarded.

2.12 DELETE FUNCTION KEY

The DELETE function key will delete the file currently under the file cursor. When the DELETE function key is pressed, the operator is asked to verify that the specified file name is to be deleted. If NO (or N) is pressed, the DELETE function is aborted and the main function key menu is restored. If YES (or Y) is pressed, the file is deleted and removed from the index display, leaving a hole in its place. This hole is filled the next time the index display is updated.

2.13 BACKUP FUNCTION KEY

The BACKUP function key will make an exact duplicate of the disk in drive 0 on the disk in drive 1. In so doing, all of the files on the disk in drive 1 (if any) are erased and replaced by the copy.

Since an exact duplicate is to be made, the disk in drive 1 must have the same "sidedness" as the disk drive 0, i.e., both disks must be single-sided or both must be double-sided. If a copy onto a disk with different sidedness is desired, either a NEW DISK must be performed followed by a COPY of each file (see sections 2.11 and 2.7) or the CODOS utilities FORMAT and COPYF must be used (see section 6.7). Since double-sided disk drives are standard on MTU-130 computers, the WORDPIC Distribution Disk is double-sided. Double-sided disks hold virtually twice as much as single-sided disks, so their use is preferred.

When the BACKUP function key is pressed, warning and informational messages are displayed. To begin the backup process, the desired source disk should be placed into drive 0 (if not already there) and the disk to receive the copy should be placed into drive 1. When the disks are ready, RETURN is pressed to start. Pressing any other key will abort the BACKUP function. A single-sided disk will require 1 1/4 minutes to backup while a double-sided disk will take 2 1/2 minutes. These times include formatting the duplicate disk, copying the files, and readback verifying the accuracy of the duplication.

When the backup is complete, the the disks may be changed as desired and RETURN pressed to continue. The index is then read from the disk in drive 0 and WORDPIC is ready for another operation.

2.14 EXIT FUNCTION KEY

The EXIT function key is used to exit from the WORDPIC shell and enter the CODOS command processor. Using the Reset key is no longer necessary to accomplish this function. If however WORDPIC was started with a NE (no exit) argument, the EXIT legend will be blank (f7 on the "other" menu) and Reset will be the only way to enter CODOS. If EXIT is pressed accidentally, WORDPIC can be re-entered immediately by typing WORDPIC and then RETURN.

3. WPEDIT

WPEDIT is the word processing editor component of the WORDPIC system. WPEDIT may be used to create and edit any kind of ASCII text file including program source and data files stored in ASCII format. Its many features however make it most suitable for creating and editing formatted English text.

3.1 STARTING WPEDIT

WPEDIT is normally initiated by pressing the EDIT function key while WORDPIC is displaying the disk file index. After a couple of questions are answered, WPEDIT is started with the selected existing file showing on the screen. See section 2.3 for details on this method of entering WPEDIT. Note that if WPEDIT edits a file on a disk that does not have a file called WORDPICSAVES.Z, it is created. See section 3.1.1 below for details.

3.1.1 Starting WPEDIT From CODOS

WPEDIT may also be started directly by a command from the CODOS operating system monitor. This command has the form:

```
WPEDIT  [<filename> [<linesize>] [B={<backupfile>}]]
                                   N
```

Where the arguments are as follows:

```
<filename>   - Name of the file to be edited or created
<linesize>   - Maximum line length
<backupfile> - Name of the backup file, N = no backup desired
```

If <u>all</u> of the arguments are omitted, WPEDIT will prompt:

```
EDIT WHAT FILENAME?
```

If the <linesize> argument is present and a new file is being created, its linesize will be set to the specified value. If <linesize> is present and it differs from the linesize specified when an existing file was created, WPEDIT will prompt:

```
DO YOU WANT TO CHANGE THE MAXIMUM LINE SIZE FROM xx TO yy (Y/N)?=
```

where xx is the file's existing line size and yy is the <linesize> argument. If no <linesize> argument is given and a a new file is being created, the default linesize of 80 characters will be used for the new file. For an existing file, the linesize which was specified when the document was created will be used as the default.

If the B=<backupfile> or B=N argument is not present, WPEDIT then prompts:

```
PLEASE ENTER NAME OF BACKUP FILE (RETURN=NONE)=
```

If no backup file is desired, simply press carriage RETURN. Entering a file name will cause the file to be copied to the specified file for backup purposes. If a name is specified which already exists, the program prompts:

```
BACKUP FILE ALREADY EXISTS.  OVERWRITE IT (Y/N)?
```

A "Y" reply will overwrite the file. An "N" reply will return control to CODOS.

During operation, WPEDIT uses a file called WORDPICSAVES.Z to store deleted text, save buffers, macros, and other information. If this file does not exist on the disk with the file being edited, it is created before WPEDIT continues. The size of this file is fixed at about 21K bytes. If WORDPICSAVES.Z already exists, it is opened and the saved information, such as previously defined macros, is made available during the edit session.

At this point WPEDIT will change the screen to the normal WPEDIT format and display the first lines of the existing document or just blank lines for a new document. After WPEDIT displays its screen, further action is independent of the method of entry.

A typical WPEDIT screen display is shown below:

```
                WORDPIC Editor  1.0   (C) Copyright 1983 M.T.U.
File= PAG_DOC.W          INSERT                  Col= 44  Line=  553  Top=  542

Processing Power" to the right of the company name.  This will be repeated at
the top of every page without having to be respecified on another codeline.

    When planning headers and footers, you should remember that the number of
standard text lines per page you specify does ÿnotÿ include the size of the
header and footer.  The total printed size of the page is therefore:

    Printed size = (LINES)+(size of header)+(size of footer)

where LINES is the current setting of the "LINES=" attribute (default=57).
If you want a standard 9.5 inch printed page but want to have a 2 line header
and a two line footer, you should therefore reduce LINES to 53.  You should also
remember that there are no blank lines printed between the header and the start
of your text or between your text and the footer, so you may want to include a
trailing blank line (or two) in the header template and a leading blank line in
the footer template, so your text won't run right into the header or footer when
printed.

    You may have wondered what happens to your footer when you have a pagebreak
on a "short" page, such as at the end of a chapter.  For example, suppose that
 JUSTPARA | FIND | MARK | CHANGE |  TABSTOP | PRTRCODE | CENTER | OTHER
```

WPEDIT divides the CRT display screen into three areas. The top three lines of the screen are the Command/Status Area (CSA). This area is used for entry of certain commands, display of status indicators, and display of prompting and error messages.

The CSA is separated from the rest of the display by a full-width horizontal line. This line is used as the tabstop display line. It displays a "tic" mark at each position where a tabstop is currently active. It also displays the current margin settings as downward pointing "carets".

Below the tabstop display line is the text window. This area is 20 lines of 80 characters each. The text window displays the text being entered or edited. It can be freely "moved about" to display any 20 lines of the file (10 lines for "wide" documents). A blinking rectangular cursor indicates the current point of interest in the text window. Text will be entered at this point when keys are pressed on the keyboard. The cursor can be easily moved about inside the text window by using the five cursor control keys in the "cursor cross" to the right of the main key array.

The bottom line of the CRT contains the function key legends. Two main menus are used with f8 (OTHER) used to switch between them. Some functions, such as MARK, have their own menus as well.

3.2.1 CSA Organization

The three-line Command Status Area is divided into a command line, a title/error pointer line, and a status/message line.

The top line is normally blank and is used to display operator commands which are not part of the actual text of the document. For example, a CHANGE command to perform a search-and-replace operation will require entry of the string to search for.

The second line of the CSA normally displays the MTU program title and copyright notice:

WORDPIC 1.0 Copyright 1983 MTU

If an error is made in a command, this line is temporarily replaced with an "up arrow" indicating the location of the error in the command.

The third line of the CSA normally displays the present status in the following form:

File=YOURFILENAME:0 INSERT Col= 23 Line= 2347 Top= 2340

The INSERT status indicator will be visible only when the INSERT key has been pressed, and is shown highlighted in reverse video. The three numbers on the right hand side indicate the current column and line number the cursor is on in the text window, and the line number currently displayed at the top of the text window. These numbers change automatically as the cursor is moved. Line numbers are adjusted automatically when lines are deleted or inserted. There are no line numbers displayed in the text window itself. In the event of an error condition, the left portion of the line will be temporarily overwritten with the error message. When the operator acknowledges the error message by pressing RETURN, the normal status display is restored.

3.2.2 Text Window Organization

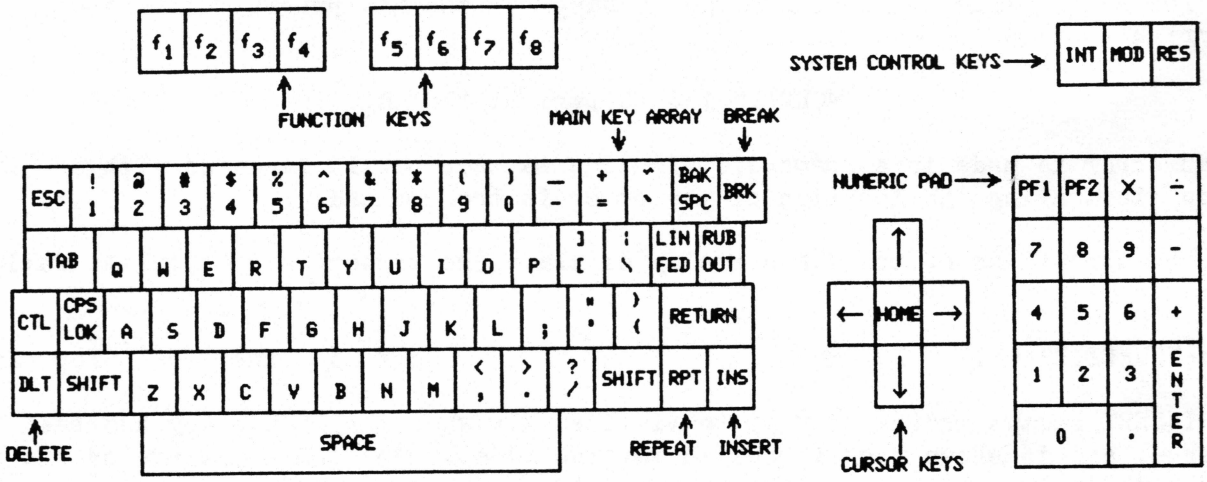
The text window displays 20 lines of up to 80 characters each. It represents a "window" into the file and can be "moved about" to display any 20 contiguous lines of the file. A blinking cursor indicates the current position in the file.

If the maximum line length is specified as greater than 80 characters when the word processor is entered, then lines will be displayed in "landscape mode". In landscape mode, two display lines are used to display one line of the file. Therefore 10 lines of the file are displayed at once instead of 20. If a given line contains less than 81 characters, it will still occupy two full display lines, with the second display line being blank. All editing operations can be performed on landscape mode lines just as for regular lines, including insertion, deletion, centering, etc. When lines entered in landscape mode are printed, they will be printed full width (assuming the printer can accept lines greater than 80 characters in length).

3.2.3 Keyboard Operation

WPEDIT uses its own keyboard driver program which makes the keyboard more suitable for high speed entry of text by experienced operators. The main improvements are "N-key rollover" and interrupt operation. N-key rollover means that new keystrokes may be recognized even if previously pressed keys are still pressed. The implementation is not perfect (certain combinations of 3 or more keys pressed simultaneously will cause a "phantom" keystroke to be recognized) but is a substantial improvement over the standard 2-key rollover driver used by CODOS. Interrupt operation means that keystrokes will continue to be recognized even though WPEDIT may be busy updating the display or accessing the disk.

Following are brief descriptions of each of the keys on the MTU-130 keyboard as used by WPEDIT:



Shift keys - Held down to enter capital letters (if Caps Lock is not engaged) and some special characters. Either shift key may be used.

Caps lock - This is an alternate action key (push-on and push-off). If latched on, all letters are entered in upper case but the number and special character keys are not affected.

Auto repeat feature - Some keys will repeat automatically if held down. These are: space, back space, rub out, delete, and the four cursor arrow keys.

Repeat - This key held down in conjunction with any other key will cause the action of that key to be rapidly repeated as long as both are held.

CTRL key - This key held down in conjunction with another key will enter a special code into WPEDIT. The various useful combinations are explained in succeeding sections and in appendix 7.1.

ESC key - Pressing this key <u>followed</u> by a special function key (such as backspace) will cause the <u>ASCII code</u> for that function to be entered literally rather than the action to actually be performed. It can be used with any special function key except carriage return.

BREAK key - Pressing this key during execution of a FIND or CHANGE command or a macro will halt execution of the command. Otherwise it does nothing.

Cursor keys - These keys are used to move the text cursor in the four indicated directions. The home key in the middle of the cross moves the cursor to the upper left corner of the text window.

Function keys - The 8 function keys correspond to the 8 legends displayed at the bottom of the screen and perform the labelled function which may change during WPEDIT operation.

Numeric pad - The number keys enter exactly like the number keys on the top row of the main keyboard except that shifting has no effect. The 4 math function keys enter the symbols normally used by computer languages (/ for ÷ and * for X). Enter is used to execute macros (section 3.16).

PF1 key - Pressing this key <u>followed</u> by any <u>two hexadecimal digits</u> (0-9 and A-F) will cause a character with that hexadecimal code to be entered (except OD which is carriage return). See appendix 7.1 for the codes of useful special characters such as Greek letters.

System control keys - These keys should not be pressed during normal operation. Refer to the MTU-130 Users Manual for their specific function.

3.2.4 Recalling Previous Commands

WPEDIT maintains a buffer of the last several commands such as FIND, CHANGE, MACRO, etc. entered by the operator. These may be recalled to the screen for re-execution or editing after an error by holding down the CTRL key and typing a B. This is denoted as CTRL/B. CTRL/B can be entered regardless of where the cursor is at the time; if it is in the text area, it will be moved to the command line and the previous command recalled there. If already on the command line, it will be cleared and the previous command recalled.

The buffer typically holds the last dozen or more commands (256 characters total) and repeated entry of CTRL/B will recall yet earlier commands in the WPEDIT session. When the buffer is exhausted, a blank line will be recalled and further recalls will wrap around to the most recently entered command again. If an undesired command is recalled, it may be cleared from the command line by pressing one of the command function keys (FIND, CHANGE, etc.) or by pressing CTRL/X. To return the cursor to the text window without executing any command, clear the command line with CTRL/X and press RETURN.

3.2.5 Error Messages

When an error occurs, the three lines of the CSA are used display the erroneous command, a pointer to the approximate location of the error, and a message describing the error condition. The error pointer on the second line is usually pointing one or two characters beyond the keyword or parameter that is causing the problem. What the pointer actually indicates is how far WPEDIT has looked into the command text before the error was detected. Nothing beyond the pointer has been looked at yet and therefore could not have contributed to the error. When an error message appears, WPEDIT will wait for the operator to press carriage return before continuing.

Following is a brief list of common error conditions. In most cases the text of the error message is sufficient to quickly pinpoint and correct the source of difficulty.

1. Misspelled command name in a code line (code lines are described later)

2. Out of range (too small or too large) or misspelled parameter in a code line

3. Incorrect syntax in an operator command such as FIND or CHANGE

4. Entering an operator command on the command line by use of the recall last command function (CNTL/B) when the function key legend for that command is not showing. The resulting message is: FUNCTION NOT PERMITTED NOW. The solution is to press OTHER (f8) to display the other menu and recall the command again.

3.3 ENTERING TEXT

To enter text, you simply type in the normal manner while the cursor is in the text window. Pressing carriage RETURN will cause the cursor to advance to the next line starting at the left margin. If the right margin is reached while typing a word, the whole word will automatically be "wrapped" to the left margin of the next line, so that no words will be broken. Note that the last character entered before the "overflow" character will cause the cursor to become "compressed" against the right margin. This is a signal that the next character entered will trigger a word wrap. If the cursor reaches the bottom of the 20 line display, the text window will automatically scroll up by one line. The top line displayed will disappear off the top of the screen (but will still be in the file), and a new blank line will appear at the bottom of the screen. This process can be repeated indefinitely. If very large amounts of text are entered (several hundred lines), the disk may be accessed momentarily at the end of a line. However, no characters will be lost and the operator can continue text entry during the disk access, which will typically last about a second. The keyboard is interrupt driven, supporting type-ahead of up to 256 characters.

3.3.1 Maneuvering the Cursor and the Text Window

The five cursor control keys located in the cursor cross to the right of the main keyboard can be used to move the cursor in the text window. The left, right, up, and down arrows will move the cursor in the indicated direction. The HOME key will move the cursor to the upper left-hand corner of the text window. The four directional keys will repeat after a short delay if held down. You may hold down more than one cursor key at once and the resulting cursor motion will be a combination of the individual key motions. Diagonal cursor motions are therefore easily accomplished. If the shift key is held down and cursor-left is pressed, the cursor moves to the left margin. Shift-cursor-right jumps the cursor to the last existing character of the line. The backspace key operates the same as the cursor left key except that it "sticks" at the left margin. Pressing cursor-down at the bottom of the text window will cause the text window to scroll up by one line. It will not advance beyond the end-of-file position. Carriage return can be used to add blank lines to the end of a file. Pressing cursor-up at the top of the display will cause the text window to "back up" one line. It will not backup beyond the beginning of the file. Holding down shift and depressing cursor-down will advance the text window to the next 20 lines of the display. Similarly, shift/cursor-up will backup by 20 lines. This is called paging through the file. Any time the cursor or display changes position, the status area of the CSA will automatically be updated. Shift/home returns to the beginning of file.

3.3.2 Overstrike Typing

Existing text may be changed by simply overtyping with the desired text. The new characters will replace the old on the screen. When overstrike typing at the end of a line, normal word wrap operation will cause a new blank line to be inserted into the file. Subsequent typing is then done on the new line instead of the next line down.

3.3.3 Deleting and Inserting Characters

The most common typing errors which are discovered immediately after pressing a wrong key can be fixed by using RUBOUT, which replaces the previous character with a blank and leaves the cursor at the position of the removed character. RUBOUT differs from backspace in that it erases a character each time it is pressed, whereas backspace simply moves the cursor without erasing.

Characters may be deleted by pressing the DELETE key on the main keyboard while the cursor is over the character to be removed. Any remaining text to the right of the cursor on the line will be "pulled over" automatically to close up the void. Holding down the DELETE key will cause repeated deletions. This makes it easy to delete whole words, etc. To prevent deleting more than was intended, type-ahead is suppressed for deletions. If the deletion is sufficiently large to make it desirable to "pull up" text from following lines to fill out the end of the line, this may be done by using the JUSTPARA function key (discussed later). Note that unlike the DELETE key, the space bar replaces characters with blanks and any remaining text to the right of the cursor will not be "pulled over". In addition, the JUSTPARA operation recognizes blanks as characters and will not remove them.

To insert characters between existing characters, the cursor is moved to the desired position and the INSERT key on the main keyboard is pressed. The CSA INSERT status indicator will appear indicating insert mode. Typing a character in insert mode will cause the character to be inserted to the left of the current cursor position. All characters to the right will automatically shift one position to the right along ith the cursor. If there is not room enough to complete the insertion on the current line, the words to the right will automatically wrap to a new line below the present line to make additional room. Typing in insert mode can be continued indefinitely. Additional lines will be inserted automatically as necessary. Pressing RETURN or any cursor movement key or backspace will exit insert mode and extinguish the INSERT indicator in the Command/Status Area. However, the RUBOUT key will not exit insert mode so as to allow simple corrections while in insert mode.

3.3.4 Deleting and Inserting Lines

Holding down SHIFT and pressing DELETE will delete the line the cursor is on. Lines following will automatically "close up" the space. To cancel the current line without closing up the gap, the CTRL key should be held down and X pressed. CTRL-X cancels the current line but leaves a blank line to type on. SHIFT-DELETE will repeat after a brief delay if held down. The last line deleted by shift-delete is saved temporarily in a special buffer and can be recovered using the RECALL key (described later).

SHIFT-INSERT will insert a blank line above the line the cursor is currently on and move the cursor to the new blank line. This key also repeats after a brief delay for easy insertion of several blank lines.

3.3.5 Entering Special Characters

WPEDIT has the capability to display and edit a large number of special characters and codes besides the usual 96 ASCII characters. Two methods are available for doing this. To enter special ASCII control codes that some printers may recognize, first press ESC and then press <u>and hold</u> CTRL while pressing a letter key. This will enter the selected ASCII control code into the file (or command line if the cursor is in the CSA) and will usually show as two hexadecimal digits displayed sideways although some codes, such as backspace, will show a distinctive shape. Control codes that correspond to keyboard keys, such as backspace, rubout, tab, etc., may be entered by pressing ESC followed by the key, except RETURN.

The other method of entry can be used to enter any 8 bit code into the file or command except carriage return. This includes the special greek, math, and line graphics characters built into WPEDIT and printable by many printers. To enter such a character, press PF1 followed by the two hex digits that identify the character. Appendix 7.1 has a table showing all 256 possible characters and their hexadecimal codes.

3.4 FIND FUNCTION KEY

The FIND function key commands WPEDIT to search for a specified line number or character string. Line numbers for a file start with line 1. The line numbers are not part of the file itself. Instead, WPEDIT keeps track of its position automatically and the line number associated with any line changes automatically as lines are inserted or deleted above it.

The FIND command can position the text window rapidly to any position in the file because it does not display the file as it searches. You can move the text window either forward or backward to any specified line. The FIND command is started by pressing the FIND function key. WPEDIT will display the word "FIND" on the top line of the Command Status Area followed by a blinking cursor. You may then type in the rest of the command, terminated by a carriage RETURN. The simplest form of the FIND command is:

```
FIND <linenumber>
```

which will display the specified line number as the first line of the text window. For example:

```
FIND 85
```

will display lines 85 through 104 of the file in the text window (lines 85 through 94 in landscape mode). If a number is specified which is larger than the last line of the file, the last line of the file will be displayed instead (if that line is blank, then nothing will be seen). Alternatively, you may specify a <u>displacement</u> from the line number the cursor is presently on by using a signed numeric argument. For example:

```
FIND +275
```

will advance the display by 275 lines from the second line number displayed at the right of the CSA. Negative displacements may be used to backup the display:

```
FIND -100
```

will backup 100 lines from the present display position. If you attempt to backup beyond line 1, line 1 will be displayed at the top of the text window instead.

In addition to searching for line numbers, the FIND command can also search for a specified character string. This form of the FIND command is:

```
FIND "<string>" [([S] [<linelimit>] [<columnlimits>])]
```

where <string> is the desired character string. The string must be enclosed in a set of either single or double quotes. The string can include blank characters and can be as long as you can type on the command line. Note however that for a string in the file to match, it cannot be split between two lines. Thus FIND is of limited usefullness in finding phrases or whole sentences.

The search for the string is always begun from the current cursor position in the text window rather than from the first line of the display and the search is always in the forward direction. The search will be continued until the specified string is found or the end of the file is reached. If the end of the file is reached, the message "NOT FOUND" will be displayed. For example:

```
FIND "this"
```

will search forward from the current line until the four characters "this" are found in the file. The text window will show this line at the top of the display, and display the cursor will be on the first character of the matching string. If the string is not found, the text window will not change and the message "NOT FOUND" appears.

You can also limit the search. This is done by specifying the optional line limit or column limit arguments in parentheses. You can specify the last line number to be searched in the file. For example:

```
FIND "solar device" (150)
```

will only search from the present cursor position up to line 150 for the string. The line limit specified must be beyond the present position or you will get an "ILLEGAL OR MISSING ARGUMENT" error message. You may also specify a positive displacement from the present position in a limited search. For example:

```
FIND "solar device" (+200)
```

will limit the search to the next 200 lines starting at the current line.

Column limits may be specified too. For example:

```
FIND 'WRITE' (C1)
```

directs WORDPIC to find the string "WRITE" starting exactly in column 1.

```
FIND 'JSR' (C8-17)
```

limits the search to strings starting in columns 8 through 17 inclusive.

```
FIND ';'(+100 C 35-)
```

searches for a semicolon anywhere from column 35 to the end of line in the next 100 lines. You must specify the "C" prefix to indicate a column limit, and if both column and line limits are given, the line limit must be given first.

Normally, FIND does not distinguish between upper case and lower case letters when matching strings. If you wish to make a distinction, the letter S should be placed just after the left parenthesis in the qualifier. For example, you could search for usage of However as the first word in a sentence in the next 1000 lines with:

```
FIND 'However' (S +1000)
```

Multiple string or line arguments may be "strung together" on a FIND command. For example, if you are presently at line 200 and wish to search for a string starting with line 1, you can do it with a single FIND command:

```
FIND  1  "100"
```

This moves the text window to line 1, without redisplaying it, and then searches forward for the string "100". This would be a way to search for a number inside a document.

For normal files WPEDIT will complete FIND commands very quickly. For large files (a thousand lines or more), WPEDIT will search at an average rate of 100 to over 500 lines per second depending on how "dense" the text is. Occasionally you may discover that you typed in the wrong string before WPEDIT finishes its search. You can abort a string search by using the BREAK key. The display will be updated to whatever point WPEDIT was at when the BREAK occurred. If however the search has already gone to the end of the file (or the limit, if specified) and WPEDIT is in the process of backing up to the original position, then the BREAK key will have no effect.

3.5 CHANGE COMMAND

The CHANGE function key on the primary menu is used to initiate global or limited search-and-replace operations. When this key is pressed, the word CHANGE will appear on the command line followed by the blinking cursor. You may then complete the command, terminating it with a carriage RETURN. The format of the CHANGE command is as follows:

CHANGE [<repeatcount>] "<string1>" [(<qualifiers>)] ["<string2>"] [Y [Y]]

The only required argument is "string1", which is the string to be replaced. Either single or double quotes can be used to enclose the string, but must be the same on both "ends" of the string. <Repeatcount> is an optional argument defining the number of occurrences of the string to be replaced, defaulting to one. Qualifiers limiting the range of the search to certain lines or columns may be specified in the same manner as for the FIND command. These qualifiers must be enclosed in parentheses. <String2> is the string which is to replace <string1>. It does not have to be the same length as <string1>. If the replacement string is long enough that it would cause the line to overflow the right margin when the substitution is made, word wrap insertion will be performed in the same manner as for normal text entry. When word wrap occurs during replacement, the display will be updated at that point, a message displayed, and the CHANGE operation terminated if a multiple replacement operation is taking place. This gives the user an opportunity to justify the paragraph before proceeding with the replacements by entering CNTRL/B. Stopping after word wrap can be suppressed however as will be described later.

Normally as each string which matches <string1> is found, it will be highlighted in reverse video, and the prompting message:

CHANGE THIS STRING (Y/N/C=CANCEL)?

will be displayed. Replying with a "Y" will cause the change to be made immediately and the search continued (if multiple replacement was specified). Replying with an "N" will leave this occurrence of the string alone but continue looking for the next occurrence. Replying with a "C" will cancel the CHANGE command at that point and leave the string intact.

If the first optional "Y" argument at the end of the command is specified, however, no prompt will be given and the number of occurrences of <string1> specified by <repeatcount> will be replaced immediately. To replace all occurrences of a string, simply specify a <repeatcount> which is very large. The command will terminate at end-of-file. If the second optional "Y" argument is also specified, there will also be no prompt when a change causes a word wrap and command execution will continue.

The CHANGE command, like the FIND command, always begins searching from the current cursor location. Some examples of search-and-replace operations using CHANGE are:

CHANGE 'us' 'MTU'

replaces one occurrence of the string 'us' with 'MTU'.

CHANGE 5 "floppy disk" "flexible diskette" Y

replaces the first five occurrences of "floppy disk" after the present cursor location with "flexible diskette", without prompting for an operator "veto" for the actual replacement but will stop if insertion of "flexible diskette" overflows the line and causes a word wrap. If instead:

CHANGE 5 "floppy disk" "flexible diskette" Y Y

were entered, all of the changes would be made immediately regardless of word wrap. At some time later it would probably be necessary to go back and rejustify any paragraphs that might have been affected.

CHANGE 100 "***" (+250)

deletes up to 100 occurrences of "***", limited to the next 250 lines. Since no replacement string was given, the string will simply be deleted. A prompt will be issued before deleting each occurrence.

CHANGE "ain't" (c40-) "isn't"

will change one instance of the specified string provided it occurs at column 40 or greater of any line.

3.6 CODE CHARACTERS

Many of the text formatting functions supported by the WORDPIC system are initiated by special code characters embedded in the text right where their effect is desired. These are called "code characters" but may also be referred to as "embedded codes". Although they are referred to as being "embedded" they are actually visible on the screen as distinctive shapes and can be edited just like any other character. Typically these are "instructions" to the printer to do something special with all of the characters that follow. Not all printers can do all of the functions specified by the available code characters however so refer to Chapter 5 for a description of what your particular printer is capable of.

Most of the code characters can be easily entered using the PRTRCODE menu. Pressing the PRTRCODE function key will display this menu which appears as follows:

UNDERbeg	BOLD beg	WIDE beg	PAUSE	SUBSCRPT	SUPERSCR		MAINMENU

This menu will remain in effect until the MAINMENU function key is pressed which returns you to the main menu. Pressing one of the other function keys will cause WPEDIT to enter a code character at the present cursor location. This code will trigger a special printer function when the file is printed. When shown on the display, a code character displays a unique shape suggestive of its meaning. The meaning of these codes is as follows:

"UNDERbeg" enters a "begin underlining" code (U). When the file is printed, this code character will not be printed but instead will instruct the printer to begin underlining. When the "UNDERbeg" function key is pressed, its displayed legend will immediately change to "UNDERend". Pressing function key F1 again will enter a "stop underlining" code (U) and change the legend for function key F1 back to "UNDERbeg". Therefore function key F1 alternately enters underline-on and underline-off when it is pressed. Returning to the main menu with f8 will always reset the legend to "UNDERbeg". All text bracketed by the begin/end underline codes, even if on multiple lines, will be underlined.

The "BOLD beg" and "WIDE beg" function keys operate in a similar fashion. "BOLD beg" enters a code (B) which will cause the printer to begin printing in bold print until a "BOLD end" code (B) is encountered. "WIDE beg" (W) causes the printer to begin printing in double-width letters until a "WIDE end" code (W) is encountered.

The "PAUSE" function key enters a "stop printer" code (P) into the text. When the PRINT program encounters this code, it will temporarily suspend printing to allow a manual operation such as a change of forms, ribbon color, print wheel, etc. The operator can then resume normal printing by pressing a function key.

The "SUBSCRPT" and "SUPERSCR" function keys are normally used for entering codes for subscripts (↘) and superscripts (↗). A "SUBSCRPT" code will cause the printer to scroll the paper up by a half line. It will continue printing at this level until another "SUBSCRPT" or "SUPERSCR" code is encountered. To print a normal superscript, you would have a "SUPERSCR" code before the superscript and a "SUBSCRPT" code after the superscript (to return to the normal line) . "SUBSCRPT" and "SUPERSCR" do not have begin/end suffixes on the function key since they may be used in various sequences. It is therefore possible to "nest" any number of sub or superscripts so that complex mathematical formulas can be printed. A subscript and superscript can be placed on the same symbol by entering a backspace (press ESCAPE and then BACKSPACE to enter a BACKSPACE code character which shows as ←) in conjunction with subscripts and superscripts.

Code characters can be freely inserted and deleted in the same manner as any other character. The displayed codes are unique from any other characters and are therefore unmistakable. Code characters are displayed on the CRT for easy editing, but are not printed on paper. Any simultaneous use of multiple embedded code operations is permitted.

Note that code characters take up space on display lines but generally do not take up space on printed lines. This may cause alignment problems if used in columnar text. Inserting additional blanks where necessary or use of the INDENT and REVINDENT commands (see section 3.7) can be used to overcome such alignment problems. Additionally, if a large number of code characters are used in any one line, there may be a noticably large amount of space at the right margin when the document is printed without justification or excessive spreading of the text if it is printed with justification. The only solution to this difficulty is to avoid redundant use of codes (such as UNDERend followed immediately by UNDERbegin) and perhaps make some slight style adjustments (such as using continuous underline rather than underline each word separately).

3.7 CODE LINES, GENERAL

There are two main kinds of embedded instruction codes in WORDPIC text files: single code characters and code lines. The single character codes were described in section 3.6. These are displayed during operation of WPEDIT as single unique characters which indicate the function to be performed, such as "begin underline". These characters are displayed so that they can be inserted and deleted during editing, but trigger special operation when the file is printed.

Code lines are similar to the single character codes, except that code lines occupy an entire line on the display instead of just one character. Although code lines occupy space on the display, they do not print or occupy space on paper. There are four different kinds of code lines, each identified by a unique symbol in the first position of the line: Tabstop (▶), Paragraph (¶), Page (↓), and Figure (▣). The reason that each of these codes occupies an entire line rather than a single character is that optional commands (keywords) and attributes may be specified on the code line. These may affect the way the printer or the display operate after encountering the code line. Since code lines can be edited just like any other line, it is easy to alter the commands and attributes at any time. Attributes set on a code line remain in effect below the code line until the end of the document, or until another code line redefines the attributes. Code lines may be SAVEd and RECALLed in the same manner as other text. Code lines always begin in column 1 regardless of the current margin settings. Code lines that are placed inside a paragraph create two paragraphs. Thus a code line cannot occur inside a paragraph.

3.8 TABSTOP CODE LINE

A tabstop code line can be created by pressing the TABSTOP function key. It is inserted before the line the cursor is on when TABSTOP is pressed and except for the distinctive identifying character (▶), is blank when created. It can then be edited to define tabstops by placing any letter, such as a T, in each column where a tabstop is desired. The tabstop code line defines the tabstops in effect on all lines after it, up to the next tabstop code line (if any). When the cursor crosses a tabstop code line, the present tabstop positions are automatically redefined. The current location of the tabstops is indicated by little "tics" on the drawn line separating the command/status area from the text window. If the cursor crosses the tabstop break line in the forward direction, then that tabstop code line defines the new tabstops in effect. If the cursor crosses a tabstop code line by "backing up", then the current tabstops will be set according to the previous tabstop code line, if any. No noticeable time is required to update these tabstops no matter how far separated tabstop code lines are in the document.

Deleting a tabstop code line will cause the tabstops of the previous tabstop code line to take effect. If all tabstop code lines are deleted, then there are no tabstops in effect throughout the document. Inserting a tabstop code line in mid-paragraph is equivalent to inserting a paragraph code line (described in section 3.9) followed by a tabstop code line.

Tabstops are useful for speeding the entry of columnar or other highly formatted text. When the TAB key is pressed, the cursor jumps to the next tabstop to the right. If the cursor is already on or beyond the rightmost tabstop, it will not move. Note that tabstops and the TAB key merely provide a way to move the cursor efficiently; once a text item is entered, there is no record of whether the operator tabbed over to the beginning position, spaced over to it, or moved the cursor there with the cursor keys. The TAB key does not store a tab code in the file.

3.9 PARAGRAPH CODE LINE

The Paragraph code line is generated by SHIFT-RETURN. The end of paragraph code line will be inserted below the cursor position, and the cursor will move to the left margin of the line following the code line. This action allows convenient use at the end of paragraphs and positions the cursor for continuation of entry at the beginning of the next paragraph. Except for the paragraph code symbol in column one, the code line will be blank when initially created.

In addition to limiting the scope of justification performed by the JUSTPARA key, the Paragraph code line can be used to specify formatting commands for the following paragraph. The available formatting commands are described in section 3.12. If there is insufficient room for all of the desired commands on one Paragraph code line, additional consecutive Paragraph code lines can be used.

The WORDPIC system will also recognize a completely blank line as a paragraph delimiter. Note that the blank line will print as a blank line as well. Multiple blank lines can also be used and they will remain in all cases until specifically deleted with shift/delete. For cases where the formatting attributes do not need to be changed between paragraphs and where the document style calls for one or more blank lines between paragraphs, use of this feature can reduce clutter on the display substantially.

3.10 NEWPAGE CODE LINE

The Newpage code line (also called a pagebreak line) is generated by pressing the NEWPAGE function key. The Newpage code line will be generated above the line the cursor is on. The Newpage code line will display a unique character in the first column and will otherwise be initially blank. The Newpage code line will not be printed out on the final document, but instead will cause the printer to begin at the top of a new page. If headers and footers have been defined, a pagebreak will print the footer, go to the top of a new page, and then print the header. Using the Newpage code line is the only way to make the printer skip to the top of the next page; the ASCII form-feed character (hexadecimal OC) is ignored by the WORDPIC PRINT program.

Like the Paragraph code line, some formatting commands can be placed on the Newpage code line. These formatting commands are described in section 3.12.

Although Newpage code lines can be placed in document file manually with the NEWPAGE function key, they are normally inserted automatically by the PAGINATE program. It is often helpful however to use Newpage code lines to separate chapters or major sections of a document or any time you want to insure that the following text starts at the top of a new page.

3.11 FIGURE CODE LINE

The Figure code line is generated by pressing the FIGURE function key. The Figure code line will be inserted above the line the cursor is currently on. The Figure code line is used to insert a graphic pixel-block image into the final printed document in place of the Figure code line. This image comes from a separate graphic file which is identified in commands on the Figure code line. Not all printers can actually print figures; see Chapter 5 for the capabilities of your particular printer. If a given printer cannot print figures, space equal to what the height of the figure would have been is skipped instead.

Figure code lines must have additional information identifying as a minimum the name of the graphic file to be printed. See section 3.12 for a description of these commands.

3.12 CODE LINE COMMANDS AND ATTRIBUTES

All of the code lines except tabstop lines can hold formatting commands as well as perform their particular function. The purpose of each formatting command on a code line is to assign an "attribute" value to a "keyword". For example, one of the available keywords is OFFSET and its attribute is a number between 0 and 25.5. PRINT uses the current value of the offset attribute to determine how far to space the printhead over from the edge of the paper before beginning to print. Other keywords and attributes are similar. In most cases attributes are numeric values but sometimes they may be a file name or one of a list of special words such as PICA used as an attribute for the TYPE keyword.

The syntax of the code line formatting commands is quite simple:

<keyword>=<value>

where <keyword> is one of the valid keywords from the list in this section and <value> is either a number between the stated limits, a special word from the list of available attribute words, or a file name. The = mark separating the keyword from the value is required. Multiple commands can exist on the same code line, separated from each other by at least one blank. Keywords and values, where they are non-numeric, may be in either upper case or lower case.

Once an attribute is assigned to a keyword in a code line command, that assignment applies to all of the text after the code line until either a different attribute is assigned in subsequent code lines or the end of the file is reached. At the begnning of the file, before any code lines are encountered, each of the keywords has a default attribute assigned to it. The value of each of these defaults is given in the following listing.

Many of the available code line commands can be used on more that one kind of code line. In the following listing, these are identified and the preferred kind of line is underlined. If a command is used on a code line that does not allow that command, an error will be generated when the cursor is moved away from the code line with the offending command. The same is true for misspelled keywords and attributes that are out of range or misspelled. Such errors should be corrected immediately before continuing.

3.12.1 LEFT=

Legal values: 1 to (right margin minus 3)
Default value: 1
Code lines used on: Paragraph, Newpage
Normally entered by: WPEDIT
Example: LEFT=15

This command is used to set the left margin on the display for each line entered and when justifying a paragraph. In PRINT, when in proportional pitch (i.e. TYPE=PROP), it is used to help insure that the left text edge is lined up correctly when printed. The LEFT margin should be considered as a mechanism to decrease the line length from the left text edge when necessary such as in outlines and narrow centered paragraphs. The normal white space margin on the printed page should be set by the OFFSET command. During text entry, the left margin may be temporarily "released" by pressing CTRL/R. Moving the cursor to another line will reinstate the margins.

3.12.2 RIGHT=

Legal values: (left margin plus 3) to maximum line length (either 80 or 160)
Default value: maximum line length of the document (80 or 160)
Code lines used on: Paragraph, Newpage
Normally entered by: WPEDIT
Example: RIGHT=65

This command is used to set the right margin on the display and is used by PRINT to determine where the right margin should be when justifying text. It is otherwise similar to LEFT= (see section 3.12.1). If the value specified is greater than the maximum line length, the right margin will be set to the maximum line length instead.

3.12.3 OFFSET= and ODDOFFSET=

Legal values: 0 to 25.5 inches in 0.1 inch increments
Default value: 0 inches
Code lines used on: Paragraph, Figure, Newpage
Normally entered by: WPEDIT, PAGINATE
Example: OFFSET=1.5 ODDOFFSET=0.5

These two commands specify an amount to offset the location of where column 1 of a line will print. PRINT maintains two offset specifications, one for even and the other for odd numbered pages (determination for odd or even is the current value of the page variable @+, see section 4.6). When used alone, the OFFSET command sets both odd and even values. The ODDOFFSET command only sets the value for odd pages. If both OFFSET and ODDOFFSET are used on the same code line, then OFFSET only sets the value for even pages.

3.12.4 INDENT=

Legal values: 0 to (right margin minus left margin minus 3)
Default value: 0
Code lines used on: Paragraph, Newpage
Normally entered by: WPEDIT
EXAMPLE: INDENT=5

This command is used to automatically indent the first line of a paragraph from the left margin. It affects the way the text is printed and, following pressing the JUSTPARA key, how it is displayed. If an INDENT command is not specified, manually created indentation (such as with TAB or with the SPACEBAR) will remain intact when JUSTPARA is used for reformatting. If an INDENT command is present, this will take control, disregard any other leading blanks, and indent the paragraph to the specified depth whenever JUSTPARA is depressed. INDENT=0 will cancel indentations until another INDENT= command is used.

3.12.5 REVINDENT= (also REVIND=)

Legal values: 0 to (right margin minus left margin minus 3)
Default value: 0
Code lines used on: Paragraph, Newpage
Normally entered by: WPEDIT
EXAMPLE: REVIND=5

This command is used to insert the specified number of spaces from the left margin on each line in a paragraph except the first line. It affects the way the text is printed, and following pressing the JUSTPARA key, how it is displayed. For example, LEFT=1 REVINDENT=5 specifies that the left margin will be at column 1, the first line of the paragraph will begin in column 1 and all lines beneath the first line of the paragraph will start 5 positions to the right at column 5 after JUSTPARA is pressed. Subsequent editing of a REVINDENTed paragraph and use of JUSTPARA will rebuild the line indentation. In the PRINT module, it has an effect only when printing in proportional mode (i.e. TYPE=PROP). In this case, the reverse indent setting is used to help make sure that the reverse indent margin will line up correctly when printed on paper. For additional details, refer to section 5.8.

3.12.6 SPACING=

Legal values: 0, .5, 1, 1.5, 2, 3
Default value: 1
Code lines used on: Paragraph, Newpage
Normally entered by: WPEDIT, PAGINATE
Example: SPACING=1.5

The SPACING command is used to set the line spacing within and/or between paragraphs. The value specifies the number of lines from the top of the line just printed to the top of the next line to move when printing. The default value is 1 which results in normal single spaced lines. A value of 2 results in double spaced lines, i.e. a blank line between each text line. If a value of 0 is specified, then lines are printed on top of each other. With a value of .5, lines are printed only a half line apart which is very useful for complex sub and superscript work such as in equation. If the value for spacing is non zero, then the actual distance on the paper is also affected by the current setting of the LPI command (Lines Per Inch, see section 3.12.7).

When determining exactly where text will appear on the page, it is important to remember that the spacing is performed after the current line is printed. This means that any extra blank space appears below the printed text line.

3.12.7 LPI=

Legal values: 0 to 8
Default value: 6
Code lines used on: Newpage, Paragraph
Normally entered during: WPEDIT, PAGINATE
Example: LPI=8

This command specifies the number of physical lines per inch to be printed and establishes the actual vertical distance the paper is moved for each line. For most printers, only 6 and 8 will be accepted. For the legal values for your printer, see Chapter 5 on PRINT.

3.12.8 PSPACING=

Legal values: 0 to 255 lines
Default value: 0
Code lines used on: Paragraph, Newpage
Normally entered by: WPEDIT, PAGINATE
Example: PSPACING=1

This command specifies the number of blank lines to be automatically inserted during printing after the end of each Paragraph code line. It is exactly equivalent to manually inserting that number of blank lines after a Paragraph code line in the file. The number of blank text lines generated is determined by the SPACING and LPI commands, just as for printed text lines. Thus if SPACING=1.5 and PSPACING=2, then there will be three blank lines printed between paragraphs. The command takes effect once the code line containing it has been processed. If it is a Paragraph code line, then that code line will be followed by the specified paragraph spacing.

3.12.9 TYPE=

Legal values: PICA or 10, ELITE or 12, 16, COND or COMP or 17, PROP
Default value: PICA
Code lines used on: Paragraph, Figure, Newpage
Normally entered during: WPEDIT
Example: TYPE=10 TYPE=PROP type=elite

This command is used to identify the desired font in which to print the following text. All of the legal values select the type style font and set the pitch in characters per inch, except PROP. Not all printers can print all of the different values selected. For those values that are available on your particular printer, see Chapter 5.

The value PROP selects the proportional font and engages the proportional printing features of the PRINT module. With this selection, the PROPPITCH command may be used to set the average number of characters per inch desired. It should be noted that proportional spacing does not imply justification, which is specified separately by the JUSTIFY command.

3.12.10 PROPPITCH=

Legal values: 0 to 25.5 characters per inch in 0.1 increments
Default value: 12
Code lines used on: Paragraph, Newpage
Normally entered during: NEWFILE, EDIT
Examples: PROPPITCH=13 PROPPITCH=10.8

This command specifies the effective pitch in characters per inch, to be used when TYPE=PROP is specified (proportional printing). Its actual effect is to specify the printed column width when proportional characters are used. For example, if the line width between the margins is 60 characters and PROPPITCH is set to 12, then the printed line width will be 5 inches, which is 60/12. In general, low values spread the characters out somewhat while high values compress the characters and may prevent the PRINT module from fitting all of the characters into the calculated column width. For the useful range of values for your printer, see section 5.8.

3.12.11 JUSTIFY=

Legal values: OFF, NONE, FLUSH, PROP
Default value: OFF
Code lines used on: Paragraph, Newpage
Normally entered during: WPEDIT
Examples: JUSTIFY=PROP JUSTIFY=OFF justify=flush

This command specifies what method of right margin justification will be used when printing. The values OFF and NONE specify that the text is to be printed with a ragged right margin, exactly as it appears on the screen. For a square right margin, specify PROP only if the TYPE is also PROP, and FLUSH if the TYPE is anything else. Flush justification will square the right margin by inserting additional blanks into the line between words. PROP (proportional) will square the margin by inserting small amounts of space between each character in the line, thus giving a more uniform appearance. In all cases, lines in the text file retain their identity when printed. PRINT does not move text from line to line when justifying. Not all printers are capable of proportional justification; see Chapter 5 for the capabilities of your particular printer.

3.12.12 PAGELENGTH=

Legal values: 0 to 25.5 inches in 0.1 inch increments
Default value: 11 inches
Code lines used on: Newpage
Normally entered during: PAGINATE, WPEDIT
Examples: PAGELENGTH=14 PAGELENGTH=1.5

This command specifies the length of the printed page in inches. This is the entire length of the page from top edge to bottom edge. When using continuous forms, it is the distance from one perforation to the next. The PRINT module uses the specified value to establish how far apart the page tops are. Normally the space occupied by the header, footer, and text body (i.e. the LINES setting) should fit within the specified pagelength. If the printing on a page goes beyond the next page boundary, the following page boundary will be used as the next top of page.

3.12.13 LINES=

Legal values: 0 to 255 lines
Default value: 57
Code lines used on: Newpage
Normally entered during: PAGINATE, WPEDIT
Example: LINES=54

This command specifies the vertical distance in 1/6's of an inch occupied by the main text body on a printed page. Although lines is the implied unit of measure, the actual unit is sixths of an inch, thus the current LPI value has no effect on this command. The text body starts at the first line below the header's last line (if present) and ends at the line just above the footer's top line. Whenever the end of text on a page is reached, the paper will be moved automatically to the bottom of the text body area (if it has not already been passed by) before printing any footer text. LINES takes effect after the Newpage code line containing it has been processed and executed, thus it affects only the pages following the code line.

3.12.14 HEAD= and ODDHEAD=

Legal values: 0 to 255
Default value: 0
Code lines used on: Newpage, Paragraph
Normally entered during: PAGINATE
Examples: HEAD=1 ODDHEAD=2

The HEAD or ODDHEAD command selects which template file page to use as the header placed above the text on a text page when printed. Setting a value of 0 means that no header should be printed. PRINT maintains two separate header template page numbers, one for even and one for odd numbered printed pages. The value tested to determine odd or even pages is the current value of the page variable, @+ (see section 4.6). When used alone, the HEAD command sets both odd and even template file page numbers. If both HEAD and ODDHEAD are specified on the same code line, HEAD sets only the template page to be used for the header on even numbered text pages.

3.12.15 FOOT= and ODDFOOT=

Legal values: 0 to 255
Default value: 0
Code lines used on: Newpage, Paragraph
Normally entered during: PAGINATE
Examples: FOOT=3 ODDFOOT=4

The FOOT or ODDFOOT command selects which template file page to use as the footer placed below text on a page when printed. Setting a value of 0 means that no footer should be printed. PRINT maintains two separate footer template page numbers, one for even and one for odd numbered printed pages. The value tested to determine odd or even pages is the current value of the page variable, @+ (see section 4.6). When used alone, the FOOT command sets both odd and even template file page numbers. If both FOOT and ODDFOOT are specified on the same code line, FOOT sets only the template page to be used for the footer on even numbered text pages.

3.12.16 FILE=

Legal values: any legal file name, optionally may include extension and drive
Default values: Newpage code line - TEMPLATES.W
Figure code line - No file (blank figure printed)
Code lines used on: Newpage, Figure
Normally entered during: PAGINATE, WPEDIT
Examples: FILE=HEAD_FOOTS FILE=FIGURE_2.G

This command is used on both Newpage code lines to specify the templates file for headers and footers and on Figure code lines to specify the file containing the graphic image. The default extension for the templates file is .W. For a graphics file the extension must be specified. The default disk drive in either case is the same drive that contains the text file.

3.12.17 @+=

Legal values: 0 to 9999
Default value: 1
Code lines used on: Newpage
Normally entered during: PAGINATE
Example: @+=50

This command allows setting the @+ variable used in a header or footer to a specific starting number. Permitted values are 0-9999 with no commas, blanks, or alphabetic characters allowed. The next page below the Newpage code line containing this command will receive the number specified. The example command above would begin the page numbering at 50. This command is useful for "delaying" page numbering in files which also contain front matter such as a table of contents.

3.12.18 @A=, @B=, ... @Z=

Legal values: see below
Default value: null
Code lines used on: Newpage
Normally entered during: PAGINATE
Example: @C=4 @T=WORDPIC @S="Description of PAGINATE usage"

These commands equate substitution phrases to the variables @A, @B, ..., @Z which are used to implement variable header and footer text. Substitution phrases including blanks must be enclosed in either single (') or double quotes (") which must be the same on both ends of the phrase, permitting the alternate mark to be used within the phrase. Substitution phrases can be up to 31 characters long. If more than 31 characters are needed, use two variables back-to-back in the template and split the phrase between them. For example a template page might contain "@X@Y" with the corresponding code line commands: @X='Why I Like To Use Long ' @y='And Wordy Document Titles'. See section 4.6 for a discussion of variable phrases.

3.12.19 DENSITY=

Legal values: 00 - 99 (see below)
Default value: Read from graphic file header, or 22 if no header.
Code lines used on: Figure
Normally entered during: WPEDIT
Examples: DENSITY=43 Density=87

The DENSITY command is used to specify the density that the graphic file specified by the Figure code line should be printed in. The density attribute consists of two digits. The first digit specifies the horizontal density in dots per inch and the second digit specifies the vertical density according to the table below:

	1	2	3	4	5	6	7	8	9	0
1st, horiz.	60	72	80	96	120	144	160	192	240	Special
2nd, vert.	48	72	80	96	108	144	192	216	288	Special

Any given printer can only print a small subset of the 100 possible combinations of horizontal and vertical density. Refer to Chapter 5 for the capabilities of your particular printer. If a density is not specified on the Figure code line, it is read from the graphic file header. If the graphic file does not have a header, a default value of 22 (72 x 72 dots/inch) is used. See section 6.11 for a discussion of graphic file headers.

3.12.20 HEIGHT=

Legal values: 0 to 2047 dots
Default value: Read from graphic file header, or if no header, 256
Code lines used on: Figure
Normally entered during: WPEDIT
Example: HEIGHT=155 height= 512

This command is used to specify the printed height in dot rows of the graphics image file specified by the FILE command on a Figure code line. The actual printed height in inches is also determined by the vertical density as specified by the DENSITY= command (see section 3.12.19). If no HEIGHT= command is specified, the height is read from the header on the graphic file. If HEIGHT= is specified and differs from the height in the graphic file header, the specified height is used with the image truncated on the bottom (file height > specified height) or padded with whitespace (file height < specified height). If the graphic file does not have a header, a default value of 256 is used. If the graphic file is too short, i.e., does not have 256 lines worth of data, the remaining portion of the calculated height in inches is skipped leaving whitespace. See section 6.11 for a discussion of graphic file headers.

3.12.21 WIDTH=

Legal values: 0 to 2047 dots
Default value: Read from graphic file header, or if no header, 480
Code lines used on: Figure
Normally entered during: WPEDIT
Example: WIDTH=240 width= 980

This command is used to specify the printed width in dot columns of the graphics image file specified by the FILE command on a Figure code line. The actual printed width in inches is also determined by the horizontal density as specified by the DENSITY= command (see section 3.12.19). If no WIDTH= command is specified, the width is read from the header on the graphic file. If WIDTH= is specified and it differs from the width in the graphic file header, the specified width is used with the image truncated at the right if the file width is greater than the specified width. If the graphic file does not have a header, a default value of 480 is used. Note that if the width value specified or assumed for a graphic file with no header is incorrect, the printed image will be distorted beyond recognition. See section 6.11 for a discussion of graphic file headers.

3.12.22 BACKUP=

Legal values: 0 to 255 lines
Default value: none
Code lines used on: Paragraph
Normally entered during: WPEDIT, PAGINATE
Example: BACKUP=17

This command instructs PRINT to back the paper up by the specified number of lines. The vertical distance moved backward for each line is determined by the current SPACING and LPI commands. Thus for example, if SPACING=2 and LPI=6, BACKUP=3 will backup one inch. Note that the entire Paragraph code line is processed before the action of backing up is performed. Thus, if the SPACING and LPI commands are changed on that code line, the distance backed up will reflect the new values rather than the previous values.

3.12.23 BACKPAGE=

Legal values: 0 to 255 pages
Default value: none
Code lines used on: Newpage
Normally entered during: WPEDIT, PAGINATE
Example: BACKPAGE=1

This command instructs PRINT to backup the paper one or more pages. Backing up over the first page always means moving back to the top of the current page. This will be the only movement performed if BACKPAGE=1 is specified. If the paper was already at the top of a page, then no movement would occur. If more than one page is specified, then each additional page after the first will move the paper back a distance equal to the current pagelength. Note that if the backpage attribute is 2 or more and single sheets are being used, the current sheet is simply fed out of the printer. Also, some printers will not allow backing up more than one page. See Chapter 5 for the capabilities of your particular printer.

If a Newpage code line contains a PAGELENGTH command in addition to a BACKPAGE command, the new pagelength does not take effect until the new top of page is reached (i.e. after the BACKPAGE is performed). This means that the current pagelength will be the old pagelength while the BACKPAGE is performed.

3.12.24 REM=

Legal values: any string of characters enclosed within " or ' quotes.
default value: none
Code lines used on: Paragraph, Newpage, Figure
Normally entered during: WPEDIT
Example: REM="DO NOT JUSTIFY THIS TABLE"

This command provides a means of including comments or editor's notes in the text file which will not appear on the printed document.

3.13 JUSTPARA FUNCTION KEY

When pressed, the JUSTPARA function key will "close up" the paragraph the cursor is in and redisplay it in the text window in its new form. Its most common use is to reformat a paragraph that has been fragmented by extensive insertions or deletions. Paragraphs are delimited by any code line or by a completely blank line. The use of JUSTPARA on tables or other non-text material will destroy the format of such material unless each line of the table is separated by a Paragraph code line or a blank line.

If the current margins are different from those in effect when the paragraph was entered or last justified, the text is "funneled" or "expanded" to conform to the new margins. Also, the justified text is made to conform to the INDENT or REVINDENT attributes currently in effect. The paragraph being justified does not have to be entirely visible on the display and may be as large as desired. When a paragraph has been justified, each line contains as many words as possible without the use of hyphenation. The resulting text is justified "ragged-right".

When the first word of a line is "pulled up" to join the end of the preceding line, one blank space is inserted. If the line being extended was terminated by ".", "?", or "!", then two blanks are inserted. Blanks within a line are not removed, since the normal operation of WORDPIC INSERT and DELETE keys does not cause extraneous blanks in midline. This allows the insertion of deliberate spacing wherever desired. Thus whole words should be removed from a document by using the DELETE key rather than typing over them with spaces.

JUSTPARA and SPLIT (see 3.14 below) are the only functions in the WORDPIC system where text is moved from line to line. Justification by the WORDPIC PRINT program, if desired, is limited to filling out lines with extra space in order to achieve an even right margin.

3.14 MARKING AND MANIPULATING TEXT BLOCKS

The MARK command is used to designate blocks of text to be deleted, copied, moved, or saved. Pressing the MARK key will display the MARK secondary menu:

SPLIT	FIND		LINE	PARA	PAGE	ACTION	CANCEL

The MARK secondary menu contains a blank function key which is reserved for future enhancements. Pressing LINE will cause the line the cursor is on in the text window to be highlighted in reverse video. This line is now marked. If LINE is pressed again, the next lower line will also be marked, thus making it easy to mark several lines. If instead the cursor is moved to a different line (either below or above the first marked line) and LINE is pressed again, all the lines between the first marked line and the second marked line inclusive will be marked. Pressing PARA will mark the entire paragraph (including any part that may be off-screen) the cursor is on. This includes the text of the paragraph plus the paragraph delimiter (blank line or code line) <u>above</u> the text. Pressing PAGE will mark the entire page the cursor is on as delimited by Newpage code lines. The SPLIT function key (which shows only if nothing has been marked) is used to split a single line into two lines. The line the cursor is on will be split with the cursor, its chacter, and all text to the right being moved down to a new line below the current line. Thus using SPLIT as a "prefix" makes it possible to effectively mark a portion of a line. The CANCEL key can be used to cancel the marking operation (removing the highlighting) and return to the main menu without action. To mark large blocks, the FIND key, which functions just like the main menu FIND key, may be used to reposition the display between MARKs.

Regardless of how it is marked, the resulting highlighted lines constitute a block of text. Pressing the ACTION key displays a menu of possible block actions:

MARKMORE	FIND		DELETE	COPY	MOVE	SAVE	CANCEL

The MARKMORE function key simply returns you to the previous MARK menu for marking additional lines before performing the desired action. The DELETE key removes the marked block, saving the deleted block in the delete buffer. If the block is too large to save in the delete buffer, WPEDIT will display the warning message:

WARNING. THIS BLOCK IS TOO BIG TO RECALL. DELETE IT (Y/N)?

Entering a "Y" reply deletes the block. An "N" reply cancels the action, leaving the text marked. The CANCEL key can then be used to cancel the marking operation if desired, or the block may be saved by writing it to a file using the SAVE function key.

Before using the COPY or MOVE function keys, the cursor should be moved to the desired destination for the copy or move. The block will be inserted above the line that the cursor is on. The FIND key can be used to locate the destination for the COPY or MOVE if desired. Pressing the COPY or MOVE function key will complete the function and return to the main menu. COPY differs from MOVE in that MOVE deletes the original block and COPY does not. The destination for a COPY or MOVE must be outside the marked block.

The SAVE function key can perform several functions and requires completion of the command by typing on the command line. If a numeric argument between 1 and 8 is specified on the SAVE command, then the marked block is placed in a save buffer of that number and can be subsequently brought back with a RECALL command specifying the same number. The saved block is saved in a special dedicated disk file called WORDPICSAVES.Z on the same disk as the file being edited. Therefore saved blocks can be recalled in another editing session, even after the computer has been turned off. If a save buffer already contains text saved there previously, it will be overwritten by the new text. If the marked block is too large to SAVE (greater than about 3000 characters), an error message will be generated.

If a file name is given, then the marked text block is written to the specified file. The file name may contain both an extension and a drive number specification if desired. The default extension if one is not specified is .W. If the file already exists, a message will be displayed giving the operator the option of overwriting the file, appending the new text to the end of the file, or cancelling the save. If there is insufficient room left on the disk for the save, an error message will be given and the save cancelled. A device name can also be specified instead of a file. For example:

SAVE P

will output the marked block to the printer. However, it should be noted that this printed output will be a literal image of the marked block, and will not provide any of the special formatting features of the full WORDPIC PRINT program. Additionally, code characters and code lines will be printed.

After the selected action is completed on the MARK/ACTION menus, the text is unmarked and the main menu is restored.

3.15 RECALL FUNCTION KEY

The RECALL function key is a multi-purpose function key which requires the operator to complete the command by typing on the command line. If no arguments are specified, then the RECALL key will cause the last line or block that was deleted to be recalled above the present cursor location. If a number between 1 and 8 is given as the argument, then the block that was last saved in the save buffer of that number is inserted above the present cursor location. Since the save buffers reside in the WORDPICSAVES.Z file on the same disk as the document being edited, they will retain saved text from one WPEDIT session to the next. If a filename is specified as the argument, then that file is inserted into the file above present cursor location. The file name may contain both an extension and a drive number specification if desired. The default extension if one is not specified is .W. The primary main menu is displayed after completion of the RECALL function.

3.16 CENTER FUNCTION KEY

The CENTER function key centers the line the cursor is currently on in the text window, between the current margins. It does this by moving the text of the line right or left until the the number of blanks between the left margin and the first non-blank character equals the number of blanks between the right margin and the last non-blank character.

If the centered text is free of embedded code characters, then it will indeed be centered when it is printed. However if it does contain code characters, such as begin/end underline, bold, wide, backspace, or sub/superscript etc., centering on the printed page will be off slightly because these code characters occupy space on the display but not on the printed page. To overcome this problem, the text of the title to be centered should be entered and centered first <u>without</u> any embedded code characters. Then the code characters should be <u>inserted</u> where they are needed. The resulting line with its codes will then no longer appear centered on the screen but it will print correctly.

A related problem can occur with wide characters. If all or any part of a line to be centered is wide, the printing will not be correct. To overcome this, first enter the text without code characters and then center it with the CENTER function key. Next insert the begin wide and end wide codes and any others where desired. Finally, delete one of the left leading spaces for every <u>two</u> characters that will be printed wide. Note that it is easy to define a line which, when printed, will be too wide to fit in the margins. If the line is too wide, its remaining part will overflow the right margin and possibly be truncated or wrap to the next line depending on the printer being used.

3.17 MACROS AND MACRO FUNCTION KEY

The MACRO function key is used to store often-used words or phrases or to "teach" WPEDIT a "new" command as a sequence of keystrokes. A macro definition allows a series of up to 31 keystrokes to be associated with a number. Subsequently pressing the ENTER key followed by the macro number will cause the list of keystrokes to be processed by WORDPIC just as though they were entered from the keyboard. This facility effectively allows you to define "new" editing operations in terms of existing capabilities.

To define a macro, the MACRO function key should be pressed. The command should then be completed on the command line and be terminated by a carriage RETURN. The format for a macro definition is:

MACRO <number> "<characters>"

where <number> is between 1 and 8 and is the macro number being defined. The characters defining the macro must be enclosed in either single or double quotes (same on both sides) followed by a carriage RETURN. The number of characters is limited to 31 or less. Defining a macro which already exists redefines the macro. For example, the following command:

MACRO 1 "Micro Technology Unlimited"

will define macro 1 as the phrase Micro Technology Unlimited. Subsequently pressing ENTER followed by 1 will enter that phrase into the text just as if it had been typed in by hand.

You may also embed special keys in the macro definition using the ESC or PF1 keys. For example, to define a macro which inserts 3 spaces at column 2 of the next line, you could press the following sequence of keys:

OTHER MACRO 3 " ESC cursor-down ESC shift-cursor-left

ESC cursor-right ESC cursor-right ESC INSERT space space

space " RETURN

Note that the ESC keystrokes do not count toward the 31 character limit. Subsequently pressing ENTER 3 will cause WPEDIT to advance the cursor to column 2 of the following line and insert three blanks.

If an error is made while entering a macro command, it can be recalled for editing by first pressing OTHER to display the secondary main menu (the one that shows MACRO as the legend for f3) and then pressing CTRL/B to redisplay the macro. The command line can then be edited and carriage return pressed to redefine the erroneous macro.

A macro may also call another macro or itself. If another macro is called, its termination will terminate the macro operation, i.e., it will not return to the calling macro. If a macro calls itself it will loop forever, until one of the following conditions occurs:

1. End-of-file is encountered during cursor-down.
2. Beginning-of-file is encountered during cursor-up.
3. An error condition occurs.
4. The BREAK key is pressed.

The BREAK key can be used to halt a "runaway" macro or to stop it manually at any desired point.

3.18 UPDATE FUNCTION KEY

WPEDIT operates using a unique editing process which allows a file to be edited in place, without resorting to scratch files or special file formats. During the editing process, part of the file is maintained in memory buffers and part is maintained in the disk file itself. As the editing session progresses, parts of the file are updated on disk and new text is brought into memory from disk. The UPDATE key causes all modifications to be written to disk so that the disk copy of the file accurately reflects the state of all operations at that point. The UPDATE key may be used periodically during lengthy editing sessions to help guard against ill effects due to a power failure or similar problem that might shutdown the computer.

3.19 QUIT FUNCTION KEY

The QUIT key is used to exit WPEDIT and return to WORDPIC if it was initiated from WORDPIC or else return to CODOS. WPEDIT updates the disk to reflect all editing operations performed on the file before returning. If initiated from CODOS, it also restores the normal display format and the function key legends which were in effect when WPEDIT was entered.

Using the QUIT function key is the only correct way to exit WPEDIT. Exiting by pressing INT or RESET will leave the text file in a damaged state if any modifications have been made during the edit session. Typically the damage will consist of a block of repeated text and another block of missing text with the blocks being several thousand characters in length.

4. PAGINATE

4.1 INTRODUCTION

The PAGINATE module of the WORDPIC Word Processing System is used to quickly and easily divide a document into pages for subsequent printing. PAGINATE is normally used after a document has been created with WPEDIT and before it is printed with PRINT. PAGINATE is told how big the pages are to be and it will divide up the document automatically. Unlike many other systems, however, PAGINATE displays every proposed pagebreak in the document, which may be accepted, rejected or moved. This gives the operator final control over the printed output format, so that "judgment calls" can be made to eliminate splitting tables or paragraphs across page boundaries or similar situations. Pagination is accomplished by inserting Newpage code lines into the document at desired points. The PRINT module begins a new page each time it encounters one of these lines. Pagebreaks can be subsequently moved, deleted, or changed either by using PAGINATE again or by using the WPEDIT editor.

PAGINATE supports a number of advanced features, some of which are summarized below:

1. PAGINATE automatically computes the initial "scissor point" for pages even if the page contains figures, changes in line spacing, equations with unbalanced sub and superscripts, etc.

2. PAGINATE displays each proposed Newpage code line at the computed scissor point which can be interactively altered both in content and position by the user.

3. PAGINATE supports automatic page numbering, resettable any number of times in the document. Compound numbers (for example, "page 3-18") can be used.

4. PAGINATE supports automatic insertion of headers and footers, which may be changed any number of times in the document.

5. Headers and footers may contain any amount of information in any format desired, including graphics (for example, a company logo).

6. Different headers, footers, and indentation (gutterspace) may be automatically provided on odd and even pages (for bound documents).

7. A text substitution facility supports variable text in headers and footers without altering the header/footer templates.

8. Pagination can be "undone" or modified at will.

9. Multi-column printing (first one column and then the next) is supported.

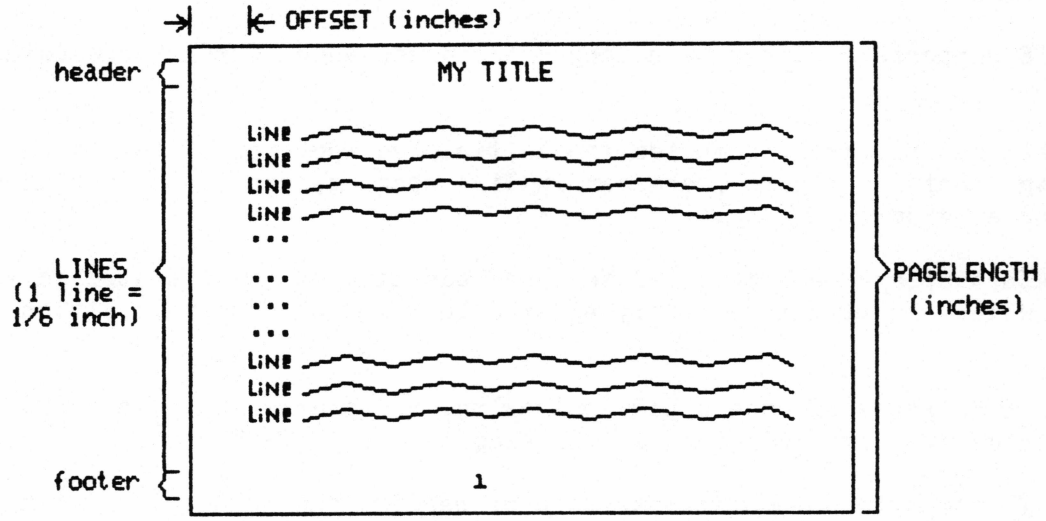
4.2 OVERVIEW OF THE PAGINATION PROCESS

After creating a document using the WORDPIC Editor and returning to the WORDPIC Shell, the operator invokes the PAGINATE module by selecting the file desired and pressing the PAGINATE function key. After receiving the answers to a couple of questions, PAGINATE will display a 19-line section of the document with a reverse-video (highlighted) page code line between two lines of the document. This indicates the proposed "scissor point" where the first page ends and the second begins.

If this proposed code line is accepted, it will be copied into the file at that point and PAGINATE will then display the next proposed new page code line position. If desired, the position of the code line may be altered by using the up-arrow or down-arrow keys to move it backward or forward in the document before accepting it, or it may be rejected altogether. The various commands that may be on the code line may also be edited. This process is simply repeated until the end of the document is reached, at which point PAGINATE is finished. The document may then be printed, if desired. These operations are covered in detail in the following sections. Headers, footers, page numbering and advanced usage (such as multicolumn pagination) is discussed after the description of the basic pagination procedure.

4.3 DEFINITIONS OF TERMS

Since PAGINATE is used to specify the layout of a document on the printed page, some basic terms will need to be defined, as shown in the figure below:



A header is information which is to be printed at the top of every page, such as the title and/or chapter number of a book. A footer is information which is to be printed at the bottom of every page. Page numbers may be included in the header or footer or both. Initially, pagination will be described assuming that no headers or footers are desired (which is the default case for PAGINATE). The OFFSET is the amount of whitespace desired (measured in inches) from the left edge of the paper to column 1 of the text of the document.

The most important measurement for PAGINATE is LINES, which specifies the amount of space to be used on each page to print the actual document text, measured in standard text lines. One standard text line is 1/6 inch. This measure is always in units of 1/6 inch on the final printed page. The default value is 57 standard text lines per page, or 9.5 inches, allowing a standard top margin of one inch and bottom margin of 1/2 inch on an 11 inch sheet, without headers or footers. Note that the value of LINES is not necessarily the same as the number of lines actually to be printed on the page. For example, if double spacing is used (by specifying SPACING=2 on a Paragraph code line), then there would only be 28 lines of text on the page, but it would still occupy 9.5 inches or 57 standard text lines. If a figure is included in the document, it might occupy 20 or more standard text lines. The value specified for LINES does not include any header or footer. PAGELENGTH is the total height of the page from top edge to bottom edge and is measured in inches. The net top margin plus bottom margin will be the PAGELENGTH minus the header minus the footer minus the inch equivalent of LINES.

As part of the initial dialog, PAGINATE is told how many standard text lines are desired per page (the default is LINES=57 when entered from CODOS, see 4.4.1 below). PAGINATE will then read the document, accumulating a count of standard text lines as it progresses, taking into account any changes in spacing, figures, lines per inch changes, etc. When it has read forward enough to match the desired value for LINES, it will stop and propose a pagebreak at this point in the document. After the proposed pagebreak is accepted, PAGINATE resumes reading until the next desired position is found and the new pagebreak is proposed. This continues until the end of the file is reached or the QUIT function key is pressed.

4.4 STARTUP DIALOG

PAGINATE is normally initiated by pressing the PAGINATE function key while WORDPIC is displaying the disk file index. After a couple of questions are answered, PAGINATE is started with the selected file showing on the screen. See section 2.5 for details on this method of entering PAGINATE.

4.4.1 Starting PAGINATE From CODOS

PAGINATE may also be started directly by a command from the CODOS operating system monitor. This command has the form:

```
PAGINATE  [<filename> [L=<lines>]  [D={Y}]  [B={<backupfile>}] ]
                                       N            N
```

Where the arguments are as follows:

```
<filename>    - Name of the file to be paginated, default extension is .W
<lines>       - Number of standard lines per page
D=Y or N      - Yes or no to delete existing Newpage code lines
<backupfile>  - Name of the backup file, N = no backup desired
```

If <u>all</u> arguments are omitted, PAGINATE will prompt:

Paginate what filename?

If the B=<backupfile> or B=N argument is omitted, PAGINATE will prompt:

Please enter name of backup file (RETURN = none)?=

If no backup file is desired, simply press carriage RETURN. Entering a file name will cause the file to be copied to <backupfile> for backup purposes. If a name is specified which already exists, the program prompts:

BACKUP FILE ALREADY EXISTS. OVERWRITE IT (Y/N)?

A "Y" reply will overwrite the file. An "N" reply will return control to CODOS.

If the L=<lines> argument is omitted, PAGINATE will prompt:

How many text lines per page do you want (excluding header/footer, if any) (RETURN=57)?=

Enter the number of lines (1/6 inch reserved for each) desired and press return.

If the D=Y or D=N argument is omitted, PAGINATE prompts:

Do you want to discard any existing pagebreaks from the file (Y/N)?=

Answering Y will cause PAGINATE to scan the file and delete any Newpage code lines it finds. Answering N will leave the file intact. At this point PAGINATE will clear the screen and show the first 20 lines of the file in its own screen format. Operation from here on is independent of the method of program entry.

4.5 THE PAGINATE DISPLAY

PAGINATE displays information on the screen in the format shown below:

```
                WORDPIC Paginate 1.0   (C) Copyright 1983 M.T.U.
Line #   1 of PAGINATEPLAY.W     Page #  0        @+=   0  ↧Line=  0 ( 57 under)
-------------------------------------------------------------------------------
↧LINES= 57
¶ TYPE=PROP OFFSET=0.7
▶   *    *                                *
                          ⌣WORDPIC PAGINATE MODULE⌣

                                INTRODUCTION
¶ JUSTIFY=PROP

    The PAGINATE module of the WORDPIC Word Processing System is used to quickly
and easily divide a document into pages for subsequent printing.  PAGINATE is
normally activated after you have created a document and before it is printed.
You tell PAGINATE how big you want your pages and it will divide up the document
automatically.  Unlike many other systems, however, PAGINATE displays every
proposed "scissor point" in the document, which you may accept, reject or move.
This gives you final control over the printed output format, so that you can
make "judgement calls" to eliminate splitting paragraphs across page boundaries
or similar situations.  Pagination is accomplished by inserting "pagebreak"
lines into the document at desired points.  The PRINT module begins a new page
each time it encounters one of these pagebreaks.  Pagebreaks can be subsequently
moved, deleted, or changed either by PAGINATE or the WORDPIC Editor, if desired.
| ACCEPT |          | BACK PG. | TEMPLATE |  | SHOW VAR | SET VAR | REJECT | QUIT |
```

The top three lines form a command/status area; the middle 20 lines form a text window, and the bottom line contains the active function key legends. The text window will be filled with the first lines of the document in the same manner as the WPEDIT display, except that a reverse video (highlighted) Newpage code line will be displayed above the first line of the document. This is the proposed pagebreak location. PAGINATE always proposes a pagebreak at the very start of the file so that headers and footers may be specified for the first page, if desired. For this initial description, headers and footers will be ignored.

The line immediately below the PAGINATE signon-message and copyright notice is the status line. The status line displays several parameter values reflecting the current state of affairs <u>if</u> the proposed Newpage code line was accepted at its current location. The leftmost part indicates the line number in the file where the currently proposed pagebreak is and the name of the file. The "Page=" display indicates the number of the page immediately <u>above</u> the proposed pagebreak. Since there is initially nothing above the break, the page number is zero. The "@+=" displays the current value of the automatic page number, which will be discussed later. The "LINE=" displays the current number of standard text lines between the last accepted pagebreak and the currently proposed pagebreak. Since initially no pagebreaks have been accepted, the value is 0. The indicator, "(57 under)", is an aid in manually positioning the Newpage code line and will be explained shortly.

The function key legends at the bottom of the screen are:

```
| ACCEPT |          | BACK PG. | TEMPLATE |  | SHOW VAR | SET VAR | REJECT | QUIT |
```

Initially we will only be concerned with the ACCEPT, REJECT and QUIT function keys. The normal keyboard keys are disabled (they just "beep" when pressed) and no flashing cursor is displayed on the screen. This is because only the function keys and cursor (up-arrow and down-arrow) keys are needed to direct the pagination process. The ACCEPT function key tells PAGINATE to insert the proposed code line into the file at the displayed position, and to advance to the proposed end of the next page. The REJECT function key tells PAGINATE to discard the proposed code line and advance to the next proposed end of page. The QUIT function key is used to cease pagination and return to the WORDPIC shell (if initiated from there) or CODOS.

The first proposed page code line (in reverse video) contains the standard WORDPIC Newpage code symbol (↧) in column 1 and "LINES=XX" after it. The "LINES=XX" command on the code line indicates the number of standard text lines per page that was specified during the startup dialog. The "LINES=" command will not appear on any following Newpage code lines unless its value is changed. Later a method for easily changing this number at any time will be described.

Since headers and footers are being ignored for the moment, there is no need to accept a pagebreak at the beginning of the document. Therefore pressing the REJECT function key (f7) will reject this initial Newpage code line. The text window will be erased and refilled with lines from further on in the document. The status line will be updated and the proposed code line will no longer appear at the top of the screen. Instead, it will appear with 12 lines of text above it and 7 lines below it (6 lines above and 3 below when paginating a "wide" document). This position allows the last few lines of the previous page and the first few lines of the next page to be viewed.

The status line now displays "LINE= 57" on the right end, and the "(57 under)" has disappeared. This indicates that there are now 57 standard text lines (each 1/6 of an inch) between the proposed pagebreak and the previous pagebreak (or beginning of the file). The left hand side of the status line probably does not show LINE# 57, however, but probably some larger number. That is because this is the line number in the file, not the measure of the number of standard text lines which would have been "printed" at this point. For example, all Paragraph code lines count as one line in the file, but since they are not printed they don't count at all toward the "LINE=" display value. On the other hand, a Figure code line occupies only 1 line in the file but might occupy 20 or more standard text lines (of 1/6 inch each) when printed.

Pressing the Up-arrow or Down-arrow keys will move the proposed Newpage code line up or down by one line on the display. The right end of the status line will indicate the number of standard text lines that this position is "over" or "under" compared with the desired amount. The word "over" is highlighted when more than the specified number of lines are on the page to warn that this indicates a situation where the printed text might run over the end of the page. The shifted arrow keys move the proposed Newpage code line by one display-page (19 lines). The proposed Newpage code line cannot be moved beyond any other existing pagebreaks (if any), or beyond the end of the file. Normally the proposed break will be moved a little in the "under direction" in order to avoid splitting a paragraph across pages.

When PAGINATE reaches the end of the document, it will propose a final pagebreak. Normally this will not be accepted, but instead of pressing REJECT, press QUIT. The following prompt then appears:

Do you want to ACCEPT the proposed pagebreak before quittng (Y/N)?=

Reply "N", and PAGINATE will return to the WORDPIC Shell (or CODOS) without installing the final new page code lien. The operator may also quit at any time before the end of file, in which case it might be desirable to ACCEPT the final proposed code line.

When paginating a document with pagebreaks already in it and not discarded, PAGINATE will re-propose the Newpage code lines it finds at the position it finds them, as well as proposing new pagebreaks where applicable. This is desirable, for example, when paginating a document which presently has pagebreaks at the end of the chapters only or one in which only a few of the pages have been edited severely enough to affect pagination.

4.6 HEADERS, FOOTERS, AND PAGE NUMBERS

PAGINATE supports a simple, powerful, and flexible method for automatically generating headers, footers and page numbering on the final printed document. Headers and footers do not appear in the document file, but are instead stored in a separate file called a template file. Only one copy of the header and/or footer desired needs to be in this template file. To get the header or footer printed on the final document, it is "called up" once on a Newpage code line in the document, and then it will automatically be printed on every page. Therefore the headers and footers on all the printed pages can be changed later merely by changing the one copy in the template file and reprinting. This process is very similar to the way Figure code lines call up separate graphic files for printing.

4.6.1 Templates File

Creating a template file is very easy; it is simply typed in using WPEDIT. This may be done either before or after using PAGINATE. To create a header or footer, the desired text is typed in the horizontal position it should appear on the printed page and terminated with a Newpage code line (using the NEWPAGE function key on the OTHER menu of WPEDIT). Often the header or footer will only be a single line, but it can be more. Up to 255 headers and footers may be put in a single file, separated by Newpage code lines. For example, suppose a header and a footer were needed for the CODOS section of the MTU-130 manual. A file called M130_TPLTS could be created using WPEDIT that looked like this:

```
Chapter 4                    MTU-130 USER MANUAL                    CODOS Usage
↧
                                  page @+
↧
```

This defines a header and a footer of one line each. There is no inherent distinction between a header and a footer; either one could be used at the top or either one at the bottom of the page, depending on how they are called up. Additional headers or footers for additional chapters could be prepared by simply generating more text, with each one separated with a Newpage code line.

4.6.2 Page Variable

The "@+" in the second template is a special symbol called the page variable. When the WORDPIC PRINT module actually prints the document, it will automatically substitute the correct page number in place of the "@+" in the footer. When the footer for the first page is printed, "1" will be substituted for "@+"; "2" will be substituted on the second page footer, etc. If page numbering is not desired, the "@+" should simply be omitted from the template. The page number will appear on the printed page in whatever position the "@+" is put. It can be put in both the header and footer, at both the left and right edges, or however desired.

4.6.3 Calling Up the Templates

Headers and footers are called up by commands on Newpage code lines in the main document. For example the code line:

```
↧ LINES=57 FILE=M130_TPLTS HEAD=1 FOOT=2
```

specifies that the first page of the file M130_TPLTS.W should be used as a header, and the second page should be used as the footer. This would be the way to call up the sample templates defined in section 4.6.1. Once a HEAD= or FOOT= command is specified on a code line, it remains in effect until changed. Therefore a header/footer only needs to be called up once, and it will be printed on all subsequent pages. To cancel a header or footer altogether, specify HEAD=0 or FOOT=0.

These commands may be added to the Newpage code lines either by WPEDIT or PAGINATE. Often it is most convenient to specify the headers and footers during pagination. To call up a header or footer on the currently proposed code line, the SET VAR ("set variables") function key should be pressed. A copy of the proposed Newpage code line will be shown in the top line of the command/status area (not in reverse video), with the cursor at the end of the line. Any desired attributes that are legal on Newpage code lines may now be typed in. Pressing RETURN updates the proposed code line in the text window area.

4.6.4 Setting the Page Variable

In addition to calling up headers and footers, the value of the page variable, "@+", may also be set. For example, if numbering of the next page should start at 50, the following could be specified:

```
↧ LINES=57 FILE=M130_TPLTS HEAD=1 FOOT=2 @+=50
```

The value shown after the "@+=" on the status line display changes as pagebreaks are accepted. This is the current value of the page number variable. This value would be printed out in a header/footer on the page before the currently proposed pagebreak. It is important to realize that the values displayed on the status line are the values in force before the currently proposed pagebreak. This is because the Newpage code line is only "proposed" and is not actually in effect. It also provides a chance to view both the current value (on the status line) and any proposed changes (on the proposed line itself). All commands or variables specified on the code line apply to the page beginning with the code line, and do not take effect until the code line is accepted. Thus the footer for the page beginning with the code line above will have a page number of 50 printed, as would be expected.

4.6.5 User-Defined Variables

In addition to the page variable @+, there are 26 other user-defined variables called @A, @B, @C, and so on up to @Z. These behave in a similar manner to the page variable @+, except that the values assigned to the variables do not change unless explicitly altered. Suppose the template file previously generated is changed to the following:

```
 Chapter @C                MTU-130 USER MANUAL                @S
↧
                                  page @+
↧
```

If a Newpage code line such as:

```
↧ LINES=57 FILE=M130_TPLTS HEAD=1 FOOT=2 @C=4 @S="CODOS Usage"
```

is now created, the printed result will be the same as before, but now the same template can be used while still being able to change the chapter or section name at some other point in the document. For example,

```
@C=7
```

would change the chapter number to 7 from this point forward. The other attributes retain their previous values.

Variables may also be combined, for example, to create a compound page number. Suppose a template on page 3 of the template file is created which contains:

```
@c-@+
```

and a Newpage code line calling for this template is created such as:

```
↧ FOOT=3 @c=5 @+=1
```

then the printed result will be "5-1" for the new page, "5-2" for the following page, etc. Notice that either upper or lower case names (@C or @c) may be used for the variable; both are equivalent. Values assigned to the variables may be up to 31 characters in length. If the value to be assigned to the variable is to include any blanks, then it must be enclosed in quotes, otherwise, the first blank is assumed to be the end of the value. Either single (') or double quotes (") may be used, but must be the same on both ends of the character string. Only a number may be assigned to the @+ variable and it should not be enclosed in quotes.

4.6.6 The Place Holder Character (\)

Since the number of characters substituted for a variable when it is printed may be shorter, equal to, or longer than the two character variable name, remaining text on the line may be shifted left or right when the substitution is made at print time. In some cases this is desirable, and in some cases it is not. For example, consider the following header template:

```
@S                    page @+
```

The intention here is that the value assigned to @S will be the current section title, for example:

```
↧ LINES=57 head=4 @S="Editing Text"
```

The printed position of the "page" part of the footer depends on the length of the section name and thus may shift off-center for different section name lengths. To solve this problem, place a backslash (\) character in front of "page @+" in the template. The backslash is treated by the PRINT module as a placeholder. It is not printed, but instead tells PRINT that any text shifting due to substitutions should not propagate beyond the backslash. This will keep the page number in the desired place:

```
@S                    \page @+
```

4.6.7 Multi-Line Templates

Thus far, only examples of single-line headers and footers have been given. However, multiline templates are also acceptable. When planning headers and footers, it should be remembered that the number of standard text lines per page specified does not include the size of the header and footer. The total printed size of the page is therefore:

Printed size = (LINES)+(size of header)+(size of footer)

where LINES is the current setting of the "LINES=" attribute (default=57). Thus if a standard 9.5 inch printed page is desired but a 2 line header and a two line footer are also desired, LINES should be reduced to 53. Also note that there are no blank lines printed between the header and the start of the text or between the text and the footer, so it may be desirable to include a trailing blank line (or two) in the header template and a leading blank line in the footer template, so the text won't run right into the header or footer when printed.

4.6.8 Attribute Changes Within Templates

Fonts may also be changed (by using the TYPE command on a Paragraph code line within the template) or other attributes may be changed within a template without affecting the the attributes in effect in the text portion of the document. In fact, when a template is called up, nearly all of the attributes temporarily revert to their default values regardless of what they were in the text at the time. The attributes that will revert are: HEIGHT, JUSTIFY, LEFT, LPI, OFFSET, ODDOFFSET, PROPPITCH, PSPACING, REVINDENT, RIGHT, SPACING, and TYPE.

A graphic image may even be called up to be printed in a header or footer. As an example of this, suppose a header including a company logo with the company name and the title of the document beside it is desired. Assuming that the logo has been previously created as a graphic image 50 dots high using the GREDIT or ANNOTATE programs, a header template such as:

```
FILE=LOGO.G DENSITY=34
BACKUP=2 TYPE=ELITE
                                Mycompany, Inc.          @T
```

can be specified. Now if this header is called up from a Newpage code line such as:

```
LINES=53 HEAD=5 @T="Quotation 5/19/83"
```

the PRINT module will print the header at the top of the page, backup two lines, print the company name beside the logo, and then print the title, "Quotation 5/19/83" to the right of the company name. This will be repeated at the top of every page without having to be respecified on every Newpage code line.

4.6.8 More on LINES=

The reader may have wondered what happens to a footer on a "short" page, such as at the end of a chapter. For example, suppose that after pagination there are only 10 lines on the last page of chapter 1 of the document, and that chapter 2 starts on a new page. Assume further that LINES=52 has been specified and that a 3 line header (including two trailing blank lines) and a two line footer (including one leading blank line) are being used. The WORDPIC PRINT module will print this last page as follows.

At the top of the page it will print the header including the two blank lines. It will then print the last 10 lines of text. When the pagebreak is encountered, it will automatically print 42 additional blank lines (because LINES=52 was specified), and then print the footer, including the leading blank line. Thus the footer will be correctly placed regardless of how little text is on the page.

This has several benefits that may not be immediately apparent. For example, once a document has been paginated, the user can go back into the WORDPIC Editor and delete or add lines to the text without adversely affecting the position of the footer (so long as more lines than can be physically printed on the page in the specified space are not added). During the pagination process the operator could deliberately leave a page a few lines "under" to leave room for a footnote or extra sentence to be added later. Also, since page numbers are not computed until print time, new pages can be inserted into the document without having to manually renumber the pages.

4.7 SHO VAR AND TEMPLATES FUNCTION KEYS

When paginating a complex document with several variables or different headers and footers in different parts of the document, the operator may forget what the current setting of a variable is, or which template is desired. The SHO VAR function key will show the values of the currently defined variables. Remember that variables are not in force until the Newpage code line defining them is accepted, so SHO VAR will not show variables defined on the currently proposed code line. After viewing the variables, the RETURN function key is pressed to re-enable the normal PAGINATE display.

Pressing the TEMPLATES function key will cause PAGINATE to prompt:

```
Display what template file (RETURN=TEMPLATES        )?=
```

The name of a desired template file may be entered or just RETURN pressed to see the default file. If a Newpage code line with a "FILE=" command has been accepted previously, it will be the default file name instead of TEMPLATES. The text window will be cleared and the first page (template) of the file displayed, along with a new function key menu.

Pressing the NEXT PG function key will display the next template in the template file (if any). PRIOR PG backs up to the prior template (if any). A template may be selected by number by pressing FIND PG and then typing in the desired page number. The purpose of displaying the templates for headers and footers is to help refresh the operator's memory as to what they look like and what variables are used. If a template needs to be changed, however, PAGINATE will have to be exited and the WORDPIC Editor used to make the change.

4.8 BACK PG FUNCTION KEY

As mentioned earlier, a proposed pagebreak cannot normally be moved before a previously accepted pagebreak. The BACK PG function key allows this to be done. Pressing BACK PG tells PAGINATE to discard the presently proposed Newpage code line and then backup to the prior Newpage code line and re-propose it. This process may be repeated as often as needed to "undo" any amount of prior pagination. Typically the BACK PG function key would be used if it is discovered that had one more line been squeezed on a previous page then an "orphan" line at the beginning of the next page could have been avoided.

4.9 ALTERNATE-FACING PAGES

An application may call for a printed master for a report, manual or similar document where pages will be published on both sides of the paper and bound at the edge, so that two pages are visible at once as in an ordinary book. This presents two new problems, both of which are easily solved using PAGINATE. First, a different offset (whitespace) will be needed on odd and even pages, so as to leave a wider "gutterspace" where the binding is between the pages than at the edges. Second, it may be desirable to use different headers or footers on odd and even pages. For example, some books print the title on left-facing pages and the chapter name on right-facing pages. The following commands are available to simplify the achievement of these needs.

ODDOFFSET= ODDHEAD= ODDFOOT=

As their names imply, these commands specify the attributes to be used for the offset (lefthand margin whitespace), header, and footer, respectively, on odd numbered pages. Assuming that the report is to be started with a right hand page numbered page 1 (and page 2 on the back of page 1), the following page code line can be included at the start of the document:

↓ LINES=52 HEAD=1 FOOT=2 ODDHEAD=3 OFFSET=0.5 ODDOFFSET=1.5

This calls for different headers on odd and even numbered pages, but the same footer on both. Odd numbered pages (whose left edge will be at the center binding) will have a lefthand margin of 1.5 inches, and even numbered pages (whose left edge will be at the exposed edge of the book) will have a half inch left margin and presumably a 1.5 inch right margin if the text width is 6.5 inches.

4.10 PAGINATING MULTICOLUMN TEXT

One unusual feature supported by the WORDPIC system is the ability to print multicolumn text by actually printing first one column and then the next on the same page. A two-column format is frequently used for newsletters and similar documents. In other systems, this would be accomplished by printing long "galleys" of text and then "pasting-up" the columns manually. Using WORDPIC, the document can be created as a single long column and PAGINATE used to do the cut-and paste operation.

For example, assume that the first column is to be printed one half inch from the left edge of the paper and the second column 4 inches from the left edge, and that each column should be 50 lines long. If each column is considered to be a "page" for purposes of numbering, the procedure is straightforward.

To accomplish this, OFFSET=4.0 can be used to specify the right column starting position, and ODDOFFSET=0.5 to specify the left. All that remains is to get the printer to put odd and even numbered "pages" on the same physical piece of paper. To do this, BACKPAGE=1 is specified by the operator on the Newpage code line of each even numbered page, as PAGINATE divides up the "galley" into 50-line columns. When the document is printed, the first column will be printed using the ODDOFFSET value of 0.5. At the bottom of the column, the Newpage code line will send the printer back to the top of the page (because of the BACKPAGE=1 command), and the 4 inch offset will be used (because the page number, @+, is 2) to print the second column. Similarly, the lefthand column on the next sheet of paper (no BACKPAGE) will be "page" 3, and the righthand column will be "page" 4.

Note that not all printers will actually be able to perform the backpage operation required for multi-column printing. Note also that with some printers, backpage may work satisfactorily only when using cut sheets and friction feed while for others, it may only be possible when using continuous forms. Chapter 5 contains these and other operational considerations that apply to the particular printer being used.

4.11 NOTES FOR SUCCESSFUL PAGINATION

1. When using headers or footers, if the main document uses "wide" mode (160 column limit), then the template file should also be created in "wide" mode. Conversely, "normal" (80 column limit) headers and footers should only be used with normal width documents.

2. The template file cannot be the same file as the document file which calls it up.

3. A template for a header or footer <u>should not</u> have any commands on the Newpage code lines used to separate the templates in the file. Any that are present are ignored.

4. During Pagination, the SET VAR function key may not be used to change the proposed Newpage code line to any other kind of code line. The WORDPIC Editor must be used to edit the document.

5. The only proper way to exit from PAGINATE is to use the QUIT function key. Any other method (such as RESET) may cause unpredicatable damage to the file.

6. If proposed code lines are repeatedly REJECTed such that more than 255 standard text lines occur between pagebreaks, the "LINE=" portion of the status line will become meaningless.

7. Using the OFFSET= command without an accompanying ODDOFFSET command automatically sets that offset for both odd and even numbered pages. Similarly, values specified for ODDHEAD or ODDFOOT will be overruled by subsequent HEAD or FOOT specifications. Note that there is no such command as EVENHEAD, EVENFOOT or EVENOFFSET.

8. If a character string longer than 31 characters is assigned to an @ variable, it will simply be truncated to 31 characters. If more than 31 characters are needed, two variable names may be placed back-to-back in the template and the string split between them.

9. If more commands or variables are needed than will fit on the <u>first</u> Newpage code line of a document, two or more page code lines back-to-back may be used. For the first page, the PRINT module will not start a new page for each, but will instead process them as a group.

5. NEC PC-8023 PRINT PROGRAM

This chapter describes the NEC PC-8023 dot matrix printer and companion WORDPIC PRINT program. Most of the information in this chapter is only applicable to WORDPIC users who have this particular printer connected to their system. If you have a different printer, contact MTU for a Chapter 5 and WORDPIC PRINT that matches your printer.

This chapter will summarize the capabilities of this printer, provide fundamental operational information (such as paper loading and operator controls) and describe how the companion WORDPIC PRINT program operates in conjunction with this printer. Nevertheless the detailed operator's manual supplied by the printer manufacturer should be consulted for detailed operational information such as ribbon replacement and preventative maintenance.

5.1 OVERVIEW OF THE NEC PC-8023 PRINTER

The NEC PC-8023 printer is a low cost dot matrix printer with exceptional flexibility and unusually good print quality for a matrix printer in its price range. It is the printer recommended by MTU for most general purpose applications. The following list highlights its many capabilities when used with WORDPIC PRINT:

* Three different mono-spaced type sizes - The NEC printer can print 10 characters per inch (PICA or 10), 12 characters per inch (ELITE or 12) and 17 characters per inch (COMP or 17). The corresponding maximum line lengths are 80, 96, and 136 characters respectively for 8 inches.

* Proportionally spaced type - The proportional font is a completely separate font with high resolution serif type Roman character shapes. Each character is given only as much space as it needs thus giving a typeset appearance.

* Proportional justification - The full proportional justification capabilities of WORDPIC are usable on the NEC printer

* Character set - Both the mono-spaced and proportionally spaced fonts are the full 96 character ASCII set with descenders on the lower case characters.

* Print speed - 100 characters per second for PICA text, less for the other fonts.

* Graphics - Full graphic printing capability up to 1280 dots wide by 2048 dots high in horizontal densities of 80, 96, 136, and 160 dots per inch and vertical densities of 72 and 144 dots per inch.

* Emphasizing - Double width characters and enhanced (double-strike) characters and graphics. Final draft mode of PRINT uses double-strike for everything.

* Special Characters - The full Greek alphabet, a variety of mathematical symbols, and line drawing graphic characters are available. See section 5.5 for a code chart.

* Freedom of movement - The printhead can space forward and backward on command in any sequence (embedded backspaces are honored). The paper can scroll forward and backward on command in any sequence (subscript, superscript, BACKUP and BACKPAGE all function).

* Buffer - A 2000 character buffer is standard.

* Forms handling - Both cut sheet and continuous tractor-feed forms may be used. BACKUP and BACKPAGE are usable with tractor feed.

5.1.1 Operator Controls

The NEC printer has six major operator controls that are used in the normal course of operation as follows:

1. Power switch - This is a rocker switch on the lower left side of the printer case. To minimize the possibility of electrical noise affecting the computer, the printer should be turned on and off with the computer rather than being turned on immediately before use.

2. SEL button and light - Pressing this button will alternately "select" and "deselect" the printer. The associated lamp will be on when the printer is selected and able to be operated by the PRINT program. The printer must be deselected and the lamp off before the L.F. and T.O.F. buttons will function.

3. L.F. button - Pressing this button briefly will feed the paper forward 1/6 of an inch. Holding it down will cause continuous feeding of the paper. The printer must be deselected for this button to function.

4. T.O.F. button - Pressing this button will feed the paper forward to the printer's internal top of form position. Please note that WORDPIC PRINT does <u>not</u> use the printer's internal forms length logic. Therefore pressing this button after a document is printed will not necessarily position the paper to the top of a sheet.

5. Platten knob - This is used to manually position the paper vertically when the printer is not printing. When WORDPIC PRINT starts printing a document, it assumes that the paper is initially positioned <u>just below the top margin</u>.

6. Friction/pin feed lever - This must be in the "Friction" position when using cut sheet paper. When using continuous forms paper, the friction feed should normally be disengaged. When subscript, superscript, BACKUP, and BACKPAGE are used, vertical alignment can often be improved substantially with continuous forms by engaging the friction feed. There is a risk however of paper "creep" and eventual tearing when many pages are printed with the friction feed on.

7. Forms tractors - These are located <u>behind</u> the platten and are used with continuous forms paper. They provide a limited range of horizontal movement.

5.1.2 Loading Cut Sheet Paper

To load single cut sheets into the NEC-8023, engage the friction feed and then insert the sheet into the slot immediately behind the platten while turning the platten knob. If alignment is important, feed the paper about half way out, disengage the friction feed, match the two left corners together and two right corners together and then re-engage the friction feed. The horizontal position of the paper should such that the left edge lines up with the red groove in the paper bail rod if you want the OFFSET attributes to be accurate. Finally, turn the platten so that the print head is where you wish for the first line to print (assuming no leading blank lines in your header or text). The head prints so that the tops of characters are about 1/6 inch (1 line) below the top of the metal ribbon shield. Make sure the paper bail is down against the paper if the paper has been positioned high enough for it to engage. If not, lower it against the paper later after a couple of lines have been printed. The maximum cut sheet width is 9.5 inches and WORDPIC imposes a length limit of 25.5 inches.

5.1.3 Loading Continuous Forms Paper

To load continuous form paper into the NEC-8023, you must first remove the rear top cover which covers the tractors. It may require a substantial amount of pressure to remove this cover. Once the tractors are exposed, loosen them by pushing the white plasitc tabs back against the rear of the printer. The tractors should now move freely horizontally although the left tractor's movement is restricted to about 1/4 inch. Now open both tractors and position them so that the rounded pins engage the perforations in the paper and then close them. The maximum continuous forms width is 9.5 inches (9 inches between perforations) and the minimum width is 4 inches. If you are using continuous mailing labels be sure they are at least this wide, measured between the perforations.

Now feed the paper in slowly by turning the platten knob until the paper is seen in front of the platten. Make sure the feed lever is in the PIN FEED position and move both tractors and the paper sideways until the left edge of the paper (not the tear-off strip) lines up with the red groove in the paper bail. After the paper is correctly positioned horizontally, lock the tractors by pulling the white plastic tabs forward.

Finally use the platten knob to position the paper vertically so that the print head is where you wish for the first line to print (assuming no leading blank lines in your header or text). The head prints so that the tops of characters are about 1/6 inch (1 line) below the top of the metal ribbon shield. Make sure the paper bail is down against the paper if the paper has been positioned high enough for it to engage. If not, lower it against the paper later after a couple of lines have been printed or feed out the first sheet altogether and position the second one. A unique feature of the NEC printer is that its rear mounted tractors make it generally possible to begin printing on the very first continuous form rather than wasting it. You may wish to delay replacing the rear cover until it is confirmed that the paper is feeding correctly.

5.2 STARTING PRINT

To start PRINT from the WORDPIC shell, press the PRINT function key while the file cursor is pointing to the file to be printed. WORDPIC will ask which pages are to be printed and whether rough draft or final form printing is desired (see section 2.6 for a discussion of the dialog). On the NEC-8023 printer, rough draft mode will print approximately 40% faster and wear the ribbon about half as much as final mode will. In final mode there will be no visible difference between regular and bold (enhanced) characters.

WORDPIC then starts PRINT which then asks whether single cut sheets or continuous forms are to be used. In cut sheet mode, PRINT will stop at the end of every page and wait for you to put the next sheet into the printer and press a function key. In continuous forms mode, the pages specified are printed continuously, one after the other, until the print job is complete. After the paper type selection is made by pressing f1 or f3, printing will begin.

5.2.1 Starting PRINT From CODOS

PRINT may also be started from CODOS. The syntax of the command is:

PRINT [<filename> [<pagerange>] [DRAFT]]

where the arguments are as follows:

<filename> - The name of the file to be printed, default extension is .W
<pagerange> - The number of the single page to print or a pair of numbers with a dash between them defining a page number range (see below)
DRAFT - If present, prints in rough draft mode, otherwise prints in final mode

If all of the arguments are missing, PRINT will ask for the file name to be printed. After the file name is entered, PRINT will ask whether single sheets or continuous forms are to be used and will unconditionally print all of the pages in the file in final mode.

If the pagerange and DRAFT arguments are missing, PRINT will ask whether single sheets or continuous forms are to be used and will unconditionally print all of the pages in the file in final mode.

The <pagerange> argument can take one of several forms. For printing just one page, <pagerange> should be a single number such as 17. To start printing at a specified beginning page number and stop at a specified ending page number, <pagerange> should be two numbers separated by a dash (minus) such as 5-14. If there is no number after the dash, such as 10-, printing will start at the specified page number and go to the end of the document. Remember that these are sequential page numbers, i.e, a simple count of Newpage code lines encountered in the file, and are not necessarily the page numbers that will be printed on the document.

If the DRAFT argument is included on the line (either before or after the <pagerange> argument), printing will be done in draft mode. On the NEC-8023 printer, rough draft mode will print approximately 40% faster and wear the ribbon about half as much as final mode will. In final mode there will be no visible difference between regular and bold (enhanced) characters.

After analyzing the arguments, PRINT asks whether single cut sheets or continuous forms are to be used. In cut sheet mode, PRINT will stop at the end of every page and wait for you to put the next sheet into the printer and press a function key. In continuous forms mode, the pages specified are printed continuously, one after the other, until the print job is complete. After the paper type selection is made by pressing f1 or f3, printing will begin.

5.2.2 "Printer Not Responding" Message

Before printing the document, PRINT checks to see if the printer is connected to the system, is powered up, is selected, and is ready to print. If any of these requirements is not met, it will display:

PRINTER NOT RESPONDING. WHEN PRINTER IS READY, PRESS f1 TO CONTINUE.
PRESS f8 TO ABORT.

Correct the problem and press f1 to continue. The SEL light must be on and the PE light must be off for PRINT to continue. After this initial check, a non-responding printer may cause PRINT to wait indefinitely for a response.

5.3 ACTION WHILE PRINTING

Once you have made the selection between single sheet and continuous forms, PRINT begins searching for the first page to print. If the first printed page is a substantial distance into the file, this search will take some time, typically 1 to 2 seconds per page skipped. The disk activity light will be on during this time.

5.3.1 Sequential Page Number Display

Once the first printed page is found, printing will commence. PRINT will display on the screen the sequential page number of the page currently being printed. Since this is the sequential page number (see section 5.2.1), it probably will not match the page number actually printed on the document. Also, since the NEC-8023 has a moderate sized buffer, the page number displayed can get ahead of the page number being printed. Unless the pages are very short however, it will never be ahead by more than one or two pages.

5.3.2 Printing On Single Cut Sheets

When the single sheet option is selected, PRINT will pause at the end of every page and display:

END OF PAGE. PLEASE PLACE NEXT PAGE IN PRINTER AND POSITION AT TOP OF FORM.
PRESS f1 TO CONTINUE, OR f8 TO ABORT.

Because of the printer's buffer, the message may appear almost immediately after the page begins printing. Wait until the page has finished printing and been ejected. Then load the new page and press function key f1 (labelled CONTINUE) to begin printing the next page.

If the page being printed prints down to within 1.5 inches of the bottom edge, the printer's paper out sensor (the red PE indicator on the printer's front panel) may trigger prematurely. If this happens, press the SEL button to resume printing for one line. Repeat this until the last line is finished and the page ejects from the platten. Then insert the next page, align it, and press f1 to continue.

When using cut sheet paper, a couple of print format restrictions should be kept in mind. First, if the document format calls for any reverse motion such as BACKUP or BACKPAGE, it cannot be done too close to the bottom edge. Second, if such a backup goes further than to the top of the page, the sheet will be fed completely out of the printer instead.

5.3.3 Printing On Continuous Forms

When using continuous form paper, you should make every effort to avoid running out of paper while the document is being printed. If the printer does run out of paper, the bottom portion of the last page may be severely misaligned to the point of lines being printed on top of each other. This problem can be alleviated by engaging the friction feed continuously (see below) or by engaging it at the top of the last page. As with single sheets, the paper-out sensor will probably trigger before the last sheet is completely printed. Pressing the SEL button will print one additional line each time it is pressed. After the last line is printed on the page, load in some more paper. Since PRINT is in the continuous forms mode, it will not feed the sheet out of the printer after the last line is printed and it will not wait for you to to press CONTINUE before trying to print the next page. Thus if SEL is pressed one too many times, you will loose the top line of the next page.

When using continuous forms paper, a couple of print format considerations should be kept in mind. First, because the paper is somewhat loose in the tractors, there will be a noticable misalignment when subscripts and superscripts are printed. In addition, the first line printed after a BACKUP or BACKPAGE may not be aligned perfectly either. Engaging the friction feed will substantially improve alignment in these situations but there is a risk of "creep" between the friction feed and tractor feed that may eventually cause the paper to tear after many pages. This can be alleviated to a great extent by temporarily releasing the friction feed every few pages to release the accumulated creep. The tendency to creep varies considerably from one printer to the next.

5.3.4 Function Keys While Printing

Like other WORDPIC system programs, PRINT displays a menu of function key legends while it is running. You may press these to control the printing process to some extent.

The SUSPEND function key can be used to temporarily stop PRINT. Since the NEC printer has a buffer, it is necessary to press and hold the key down until an acknowledgement message is displayed. Up to several seconds may be required before the program responds. When it does respond, the function key legend changes to CONTINUE. Again, since the printer has a sizable buffer, actual printing may continue for as long as 30 additional seconds. Pressing the SEL button will not immediately stop the printer; you will just have to wait until the printer's buffer empties. In an emergency, the printer's power may be switched off. This of course would necessitate cancelling the print run (see below) and restarting.

Note that the SEL button should not be used to control stopping and starting the printer. The reason is that there is a chance that the printer will miss a character which could have a serious effect on the document format if it happened to be a special control character.

The ABORT function key can be used to permanently halt printing of the document and make PRINT return to WORDPIC (or CODOS) with an error indication. Like the SUSPEND function key, it must be held down until the program responds. After PRINT stops and returns, the printer will continue to print from its buffer for up to 30 seconds.

5.3.5 Error Conditions While Printing

While the document is being printed, PRINT can detect several different error conditions. None of these will cause printing to be aborted but each may have an unexpected or undesired effect on formatting. As each error is detected, it is displayed on the screen along with the sequential page number and line number in the text file where the error occured. The line number display makes it easy to correct the problem later with the WORDPIC editor. Note that due to the printer's buffer, the error may be detected and displayed well before its effect is observed on paper. The most common errors detected by PRINT are described in the following list:

* Illegal Command. - A misspelled command word was encountered on a code line.

* Illegal of missing value. - The attribute value specified for a command on a code line is out of range, misspelled, or incorrect in its syntax.

* Disk drive is not open - A FILE= command specified a file name with a drive number and that drive is not selected (open).

* File not found. - A FILE= command specified a file name which could not be found.

* Missing or illegal file name. - The file name in a FILE= command does not conform to the rules for file names.

* Can't proportionally space this line to fit within the margins. - Even with zero space between characters, the line was too wide to fit between the margins. Is usually caused by an unusually large number of upper case letters on the line or by wide mode characters on the line.

* Too many sequential blank lines. Limit of 128 lines was reached.

* Too many lines to backup. Limit of 128 lines was reached.

* Too many pages to backup. Would backup past first page.

* String expansion in header overflowed bounds. - When variable text was substituted for an @ variable (@A, @B, etc.) in a header, the header line became too long to fit within the right margin or the text was too long to fit between the @ variable and a "\" place holder.

* String expansion in footer overflowed bounds. - Same as above except for footer.

* Can't justify this line to fit within margins. - Even with no additional space between words, the line was too wide to fit between the margins. Is usually caused by too many wide mode characters on the line.

* Vt. density of xxx dots/inch not supported. Will use yyy. - The vertical density specified on a Figure code line or in the graphic file is not available on the NEC printer. The next highest available vertical density, yyy, will be used. The printed figure will therefore not be as high as expected.

* Hz. density of xxx dots/inch not supported. Will use yyy. - The horizontal density specified on a Figure code line or in the graphic file is not available on the NEC printer. The next highest available horizontal density, yyy, will be used. The printed figure will therefore not be as wide as expected.

* Graphics file is not printable. File is not the correct type. - The graphics header of the file name specified on a Figure code line indicates that the file is not a bit-mapped image file, i.e., the "type" byte in the header is not 0.

* Unrecognized embedded code. - A code character with no defined function for the NEC PC-8023 printer was encountered in the text and discarded.

5.4 Return After Printing

When PRINT finishes or aborts, it will return to WORDPIC (or CODOS if entered from CODOS). If any errors were encountered, the messages will remain on the screen until the operator presses RETURN to acknowledge them. WORDPIC will then display the index of the currently selected drive with the file name just printed still selected.

5.5 INTERPRETATION OF THE CODE CHARACTERS

WORDPIC PRINT for the NEC PC-8023 printer interprets code characters embedded in the text as follows:

UNDERbeg & UNDERend - All characters, including spaces, after the UNDERbeg code until an UNDERend code will be underlined. The underline will overlap the lower extremity of lower case character descenders.

BOLDbeg & BOLDend - All printing characters after the BOLDbeg until a BOLDend code will be printed more darkly than other text. If a BOLDbeg/BOLDend pair spans a figure, the figure will also be printed boldly. Bold has no effect when the document is printed in final mode since all text and figures are printed boldly.

WIDEbeg & WIDEend = All printing characters after the WIDEbeg until a WIDEend code will be printed twice as wide as normal (same height). An excessive amount of wide text on one line may cause a right margin overflow error. Wide mode has no effect on figures.

SUBSCRPT - Will cause the following text to be printed 3/8 line lower than preceeding text. The LPI value affects the distance moved but the SPACING value does not.

SUPERSCR - Will cause the following text to be printed 3/8 line higher than preceeding text. The LPI value affects the distance moved but the SPACING value does not.

Backspace ($08) - Will cause the printhead to move backward by one character width. This <u>will</u> work properly with TYPE=PROP and TYPE=16. Backspace will not back up to the left of the current OFFSET value.

Carriage return ($0D) - This is the end-of-line character used throughout the WORDPIC system. It cannot be embedded in text as a code character.

All other ASCII control codes ($00 - $1F) will cause an error message to be displayed and are then discarded. See section 5.9.2 for a listing of all printing codes ($20 - $EF) and the characters they print.

If underline mode, bold mode, or wide mode is not turned off on a line, it will carry over to the next line. In the case of underline and wide modes, the effect <u>is not applied to any leading blanks</u> on the line, even if the leading blanks extend beyond the left margin. Also, underline mode, bold mode, and wide mode operate in "push down stack" fashion. This means that if two UNDERbeg codes occur, without an intervening UNDERend code, it will take <u>two</u> UNDERend codes to turn underlining off. The same applies to bold and wide mode.

5.6 INTERPRETATION OF THE CODE LINES

This section describes how WORDPIC PRINT for the NEC PC-8023 treats each of the four kinds of codelines.

5.6.1 Paragraph Code Lines

The Paragraph code line is used by the PRINT module to mark the end of a paragraph. A completely blank line in the file will also be interpreted as blank Paragraph code line. If the line which preceeds it is a printable line, that line is always printed unjustified, as you would expect for the last line of a paragraph. This implies that you should not insert other code lines, such as a Tabstop code line, between the last line of a paragraph and the Paragraph code line. If that is done, the last line of the paragraph will be justified (assuming that justification was enabled with a previous JUSTIFY=FLUSH or JUSTIFY=PROP command) and may become quite spread out if it has only a few words on it.

The Paragraph code line provides a line on which to specify new commands for the next paragraph. The code line itself will not occupy any space on the printed page. However, if the SPACING and PSPACING attributes are both non-zero, the specified amount of spacing will occur once the code line has been processed. If blank lines are used as paragraph separators, they will be printed one-for-one and the PSPACING attribute will have no effect.

5.6.2 Newpage Code Lines

The Newpage code line is used to mark the end of a page. It is used by the PRINT module to signal when it is time to print the footer, move to the next page, and print the header for the next page. Commands on the Newpage code line are not processed until the footer, if any, is printed. If commands specifying new values for PAGELENGTH or LINES are present on the code line, they will not take effect until the paper has been moved to the next top of page. This will be the next physical page unless a BACKPAGE was specified. In this case, the next top of page may be the top of the current physical page, or some previous page. Once this top of page is reached, new values for PAGELENGTH or LINES take effect.

When using footers, it is not necessary that a Newpage code line be the last line in the file to get the footer on the last page printed. PRINT will detect end-of-file and do this for you. However, if a Newpage code line is at the end of the text but is not the very last line in the file (i.e., followed by one or more blank lines before end-of-file is reached), there will be a blank last page printed complete with header and footer, if any. If this is seen, use the WORDPIC editor to delete such blank lines at the end of the file. When no trailing blank lines remain, the editor's cursor will not be allowed to go below your final Newpage code line.

Although Newpage code lines are treated as paragraph boundaries in WPEDIT, they are ignored for purposes of justification by PRINT. Thus the line before a page code line will be justified unless there is a Paragraph code line or a blank line between it and the Newpage code line. This makes it possible for PAGINATE to split paragraphs between pages and have them printed correctly.

5.6.3 Figure Code Lines

Figure code lines will cause a figure to be printed since the NEC printer supports graphics. See section 5.6.x for a listing of the available densities. With respect to vertical spacing before and after the figure, the PRINT module will treat the figure as if it were one line of text. This means the distance from the bottom of the previous text line to the top of the figure will be the same as that which would occur if the Figure code line were replaced with a line of text. The same thing applies to the distance from the bottom of the figure to the top of the next text line.

When chosing the height of a figure, it will often be desireable to have the figure occupy an amount of space equivalent to a integral number of text lines. This may be essential if a BACKUP will be done to backup over the figure. To calculate the height equivalent to a given number of text lines, use the formula appropriate for the current LPI and vertical DENSITY attribute settings:

```
HEIGHT = ( ( N - 1) * (SPACING * 12) ) + 8        for LPI=6, Vdens=2 (72 DPI)

HEIGHT = ( ( N - 1) * (SPACING * 24) ) + 16       for LPI=6, Vdens=6 (144 DPI)

HEIGHT = ( ( N - 1) * (SPACING * 9) ) +  8        for LPI=8  Vdens=2 (72 DPI)

HEIGHT = ( ( N - 1) * (SPACING * 18) ) +  16      for LPI=8  Vdens=6 (144 DPI)
```

where N is the number of lines. For example, to occupy 3 lines of text with LPI=6 and vertical density=2, specify HEIGHT=32. Remember that if the height is explicitly specified on the Figure code line, it takes precedence over the actual height in the graphic file. If it is less than the file height, the figure will be truncated at the bottom. If it is greater than the file height, whitespace will be added at the bottom of the figure.

5.6.4 Tabstop Code Lines

PRINT does not process Tabstop code lines in any way. Their only effect on PRINT is that if one is placed between the last line of a paragraph and a Paragraph code line, justification of that last line <u>will</u> <u>not</u> be defeated.

5.7 INTERPRETATION OF THE CODELINE COMMANDS

Most of the commands found on code lines are ultimately processed by PRINT. The following sections describe the action PRINT takes on those commands where there is likely to be some difference among the various versions of PRINT for different printers. Differences that do exist are due to varying printer capabilities.

5.7.1 OFFSET=, LEFT= and RIGHT=

PRINT uses the OFFSET (and ODDOFFSET) attribute to determine where column 1 of the text should begin printing with respect to the edge of the paper. Only 1 digit beyond the decimal point is processed thus 1.28 is interpreted as 1.2 inches. The absolute maximum offset value is 7.9.

PRINT uses the LEFT attribute to determine the location of the left edge of the actual printed text with respect to the column 1 location calculated from the offset. For the 10 (pica), 12 (elite), 16, and 17 (comp or cond) TYPE= specifications, the left text edge will be (LEFT-1)/P inches to the right of the column 1 location where P is one of the just mentioned pitches in characters per inch. When TYPE=PROP, the PROPPITCH= attribute is used instead for P. The RIGHT attribute is processed in a similar fashion.

If JUSTIFY=FLUSH or JUSTIFY=PROP, then the text on each line will be compressed or exapnded as necessary to exactly fill the space between the left text edge and right text edge calculated above. If something occurs (such as the use of wide mode characters) that causes text to be printed outside the calculated right margin, an error will be signalled.

For the NEC PC-8023 printer, the absolute maximum line length is 8 inches. Thus the sum of the the RIGHT= attribute divided by the pitch and the offset cannot exceed 8.0. Attempting to exceed this right margin limit will cause line wraparound as the printer hits the right hand stop which in turn causes a complete loss of format.

5.7.2 JUSTIFY=

The JUSTIFY attribute specifies what method of right margin justification should be used, if any. The values OFF and NONE specify that the text should be printed with a ragged right margin, spaced exactly as it is in the text file. For a justified right margin, you should specify PROP if the TYPE is also PROP, and FLUSH if the TYPE is something other than PROP. Flush justification will justify the right margin by inserting additional blanks into the line. Proportional (i.e. PROP) justification will justify the line by inserting the right amount of space between each character in the line, giving a more uniform appearance. See also the table in section 5.10 for the valid combinations of TYPE and JUSTIFY.

5.7.3 TYPE=

The TYPE command is used to identify the desired font in which to print the text. All of the legal values, except PROP, also set the pitch (i.e. characters per inch) in addition to selecting the font. The values PICA and 10 both select the pica font which prints 10 characters per inch. The values ELITE and 12 both select the elite font which prints 12 characters per inch. The values COND, COMP, and 17 select a condensed font which prints 17 characters per inch. The value 16 selects the proportional font but doesn't invoke any proportional justification. As a result the characters will be printed with varying widths. This means that the number of characters per inch will vary, but on the NEC-8023 will average out to about 16 characters per inch for normal text.

The value PROP selects the proportional font and engages the proportional printing features of the PRINT module. At this point, the PROPPITCH attribute may be used to set the column width desired. If not specified, the default is 12 characters per inch. See section 5.6.4 for a discussion of PROPPITCH. It should be noted that proportional spacing does not imply justification, which is specified separately by the JUSTIFY attribute.

5.7.4 PROPPITCH=

The PROPPITCH attribute specifies the desired average pitch (in characters per inch) to be used during proportional printing (i.e. TYPE=PROP). What is actually being specified is the column width. For example, if LEFT=10 and RIGHT=75, the printed text column will be 66 characters wide. If PROPPITCH=12, then the intercharacter spacing will be such that the printed column will be 66/12 or 5.5 inches wide.

The PROPPITCH attribute may be specified to 1 decimal place, with any additional decimal places being ignored. Thus 13.33 is treated as if it were 13.3. For the NEC-8023, a useful range of values is from 10 to 13.3. Values less than 10 will result in characters which are excessively spread out. Values much greater than 13.3 may cause the PRINT module to be unable to fit a significant percentage of the lines within the specified margins.

5.7.5 LPI=

The LPI command specifies the number of physical lines per inch the printer should print. The NEC 8023 only supports 6 or 8 lines per inch. Any other number specified is treated as if it were 6, except for 0 which is ignored. The LPI attribute establishes the actual vertical distance the paper is moved when a line feed (or half line feed) is performed.

5.7.6 LINES=

The LINES command specifies the vertical distance (in lines at 6 lines per inch) occupied by the text window portion of a page. The text window portion starts at the top of the first line below the header and ends at the top of the footer. Whenever the end of a page is reached, the paper will be moved to the bottom of the text window (if it has not already passed by) before printing a footer.

The LINES attribute takes effect after the Newpage code line has been processed and executed. This means it will affect only the following pages, and not the one being printed when the Newpage code line is encountered. If the LINES attribute is placed on a code line other than the Newpage code line, it will be ignored.

5.7.7 PAGELENGTH=

The PAGELENGTH command specifies the length of the page (in inches). The pagelength is used by the PRINT module to establish how far apart the tops of each page should be. Normally the space occupied by the header, footer, and text window (i.e. the LINES setting) should fit within the specified pagelength. If the printing on a page goes beyond the next page boundary, the following page boundary will be used as the next top of page.

The top of form logic built into the printer is not used by PRINT. The position of the paper when PRINT starts is the effective top-of-form position and succeeding tops-of-form will be an integral number of pagelengths away from that point. The maximum pagelength is 25.5 inches.

5.7.8 BACKPAGE=

The BACKPAGE command instructs the PRINT module to backup the paper one or more pages. Backing up over the first page always means moving back to the top of the current page. This will be the only movement performed if BACKPAGE=1 is specified. If the paper was already at the top of a page, then no movement would occur. If more than one page is specified, then each additional page after the first will mean moving the paper back a distance equal to the current pagelength set by the PAGELENGTH attribute.

If a Newpage code line contains a PAGELENGTH command in addition to a BACKPAGE acommand, the new pagelength does not take effect until the new top of page is reached (i.e. after the BACKPAGE is performed). This means that the current pagelength will be the old pagelength when the BACKPAGE is performed.

PRINT will not allow you to backpage before the first page it has printed. This may be a consideration when only a portion of the document is being printed and there are any BACKPAGE= commands with arguments greater than 1. The advantage of BACKPAGE is that it can be used to backup to the top-of-form without knowing how far that actually is.

5.7.9 BACKUP=

The BACKUP command instructs the PRINT module to backup the paper the specified number of lines. The vertical distance moved backward for each line is determined by the current SPACING and LPI attributes. It is important to note that the entire Paragraph code line containing a BACKUP= command is processed before the backing up is physically performed. This means that if the SPACING and LPI attributes are changed on that code line, the backup distance will reflect the new values rather that the previous values. If the BACKUP command is placed on a code line which is not a Paragraph breakline, it is ignored.

5.7.10 SPACING=

The SPACING command specifies the number of lines from the top of the line just printed to the top of the next line. The default value is 1 which results in single spaced lines. A value of 2 results in double spaced lines, i.e. a blank line between each printed line. If a value of 0 is specified, then no line feeds are sent and lines are printed on top of each other. The only values accepted for SPACING are 0, .5, 1, 1.5, 2, 2.5, and 3. If the value for spacing is not zero, then the actual distance the paper is moved is also affected by the current setting of the LPI attribute. Spacing is performed <u>after</u> the current line is printed.

5.7.11 PSPACING=

The PSPACING command specifies the number of blank lines to be printed at the end of paragraphs terminated by a Paragraph code line. It is exactly equivalent to manually inserting that number of blank lines after a Paragraph code line in the file. Thus the amount of vertical space occupied by these blank lines is determined by the SPACING and LPI attributes. When blank lines are used to separate paragraphs rather than Paragraph code lines, the paragraph spacing is controled by those blank lines; PSPACING is not invoked.

The PSPACING attribute takes effect once the code line containing it has been processed. If it is a Paragraph code line, then that Paragraph code line will be followed by the specified paragraph spacing.

5.7.12 DENSITY=

The DENSITY command specifies the horizontal and vertical density of graphic images. If a density is not specified on the Figure code line, PRINT attempts to read it from the graphic file header. If the graphic file does not have a header (i.e., it is a memory block file created by a CODOS or DMXMON SAVE command), then the horizontal density defaults to 80 dots per inch and the vertical density defaults to 72 dots per inch. The graphic densities available on the NEC PC-8023 printer are as follows:

DENSITY=32	80 DPI horizontal, 72 DPI vertical
DENSITY=42	96 DPI horizontal, 72 DPI vertical
DENSITY=62	136 DPI horizontal, 72 DPI vertical
DENSITY=72	160 DPI horizontal, 72 DPI vertical
DENSITY=36	80 DPI horizontal, 144 DPI vertical
DENSITY=46	96 DPI horizontal, 144 DPI vertical
DENSITY=66	136 DPI horizontal, 144 DPI vertical
DENSITY=76	160 DPI horizontal, 144 DPI vertical

If the specified density is not available, the next highest available density is used. If the specified density is greater than the highest available, the highest available is used. This adjustment is made to the horizontal and vertical portions of the density attribute separately; there is no attempt to optimize the aspect ratio.

5.7.13 WIDTH=

The WIDTH command is used to specify the width (in number of dots) of the graphic image in a file to be printed by a Figure code line. If a width is not specified on the Figure code line, PRINT attempts to read it from the graphic file header. If the graphic file does not have a header (i.e., it is a memory block file created by a CODOS or DMXMON SAVE command), then the width defaults to 480. If both a WIDTH command is used and the file has a graphic header and the width values differ, the file header value controls how the file is read and the WIDTH= command determines the actual printed width with the image truncated or whitespace added on the right as necessary.

The width is used to determine how many bytes to read from the file for each line of dots to be printed. The width will be divided by 8 and then rounded up to the nearest whole integer to determine the number of bytes. The actual printed figure width however will be the exact value specified. Though 2048 is the maximum value that will be accepted without error, the NEC 8023 is only able to print figures up to 1280 dots wide in its highest horizontal density mode with OFFSET=0.

There is no checking for overflow of figures beyond the printer's right mechanical stop which is 8 inches to the right of the left page edge. If a figure overflows, it will "wraparound" at the right and thus destroy the format of the current <u>and all succeeding pages</u>.

5.7.14 HEIGHT=

The HEIGHT command is used to specify the height (in number of dot rows) of the graphic image in a file to be printed by a Figure code line. If a height is not specified on the Figure code line, PRINT attempts to read it from the graphic file header. If the graphic file does not have a header (i.e., it is a memory block file created by a CODOS or DMXMON SAVE command), then the height defaults to 256. If both a HEIGHT command is used and the file has a graphic header and the height values differ, the HEIGHT command determines the actual printed width with the image truncated or whitespace added on the bottom as necessary.

The height is used to determine how many lines to read from the file for the figure. If the file is shorter than the product of width and height implies, the remaining portion of the figure will be blank.

5.7.15 INDENT=

The INDENT command specifies how far to indent the first line of a paragraph from the left margin. It is used only by the WORDPIC Editor module when justifying a paragraph. It is has no function within the PRINT module.

5.7.16 REVINDENT=

The REVINDENT command is used to specify a reverse indention amount from the left margin for each line of a paragraph after the first one. It is used primarily by WPEDIT to insert the right number of extra leading blanks on each line when justifying a paragraph. In the PRINT module, it has an effect only when printing in proportional mode (i.e. TYPE=PROP). In this case, the reverse indent setting is used to help make sure that the reverse indent margin will line up correctly when printed on paper. For additional details, refer to the section on proportional printing.

5.8 PRINTING HEADERS AND FOOTERS

The PRINT module provides for optionally printing a header and footer on each page. The text for the headers and footers will be read from a separate file, and thus does not need to be part of the main file. The source for the header and footer text will default to the file TEMPLATES.W on the same disk as the main file. This source file may be changed by specifying a FILE= attribute on a Newpage code line. If the specified file, or TEMPLATES.W isn't present on the disk, then headers and footers will not be printed.

Prior to printing each header and footer, all of the attributes which control printing characteristics are reset to their default state. These attributes are: HEIGHT, JUSTIFY, LEFT, LPI, OFFSET, ODDOFFSET, PROPPITCH, PSPACING, REVINDENT, RIGHT, SPACING, and TYPE. This means you must specify the desired values for these attributes within each header or footer, if the default value is not suitable. If there are too many commands to fit on one Paragraph code line, a second Paragraph code line may be placed immediately following the first to hold any additional commands. Once the header or footer has been printed, these attributes are restored to the values that were in effect prior to the printing of the header or footer.

Before each line of a header or footer is printed, the line will be searched for "@" variables. If found, the "@" variable is deleted from the line and the current string assigned to that variable inserted in its place. In cases where the string to be inserted will not fit within space provided, an error message will be displayed and only as much as will fit is inserted.

5.9 PROPORTIONAL PRINTING

Specifying TYPE=16 or TYPE=PROP selects the proportional font in the NEC PC-8023. In the proportional mode, a variable amount of space can put between characters, allowing for a greater average density of characters and a more even appearance. TYPE=PROP differs from TYPE=16 in that the variable spacing capabilities of the printer will be utilized. When TYPE=PROP is specified, the attribute PROPPITCH comes into play. The PROPPITCH attribute is used by the PRINT module to calculate where the beginning and ending points of proportionally printed lines should fall. Once these are determined, the characters in the line will be printed evenly spaced within these boundaries.

5.9.1 Printing of Reverse Indentation

A majority of the time, the REVINDENT attribute will be set to zero. In this case, the beginning point is established by the LEFT and PROPPITCH attributes if LEFT is not zero. For unjustified lines, the end point is established by the number of characters in the line and the PROPPITCH. For justified lines, the end point is established by the RIGHT and PROPPITCH attributes.

When the REVINDENT attribute is set to a non-zero value, then an additional "margin" is defined, henceforth called the reverse indent margin. This margin falls the specified number of characters to the right of the left margin. Most lines in a reverse indented paragraph will have blanks in the line from column one to the reverse indent margin. For these lines, the reverse indent margin is used to establish the beginning point of the line.

However, there may be lines which have text between the left margin and the reverse indent margin, such as the first line in a reverse indented paragraph. These lines are given special handling in the PRINT module. If the character immediately preceeding the reverse margin column is a space, the PRINT module will proportionally space the line in two parts. The first part will contain the characters between the left margin and the reverse indent margin. These characters will be printed using beginning and end points based on these margins. The second part consists of the text in the remainder of the line. This part will be printed using beginning point based on the reverse indent margin. The result is that this second part will fall in the same place as it would, had the preceeding part not contained any text. If the character immediately preceeding the reverse indent margin is not a space, the PRINT module assumes that it is part of a word which crosses the reverse indent margin. In this case, it does not try to proportionally space the line in two parts. Instead it proportionally spaces the line as a whole, using the left margin as the beginning point.

5.9.2 Proportional Overflow

Since the widths of proportional characters vary, it is possible to have such a preponderance of wide characters on a line that it is not possible to physical fit the line between the margins, even if the space between characters is reduced to zero. This is most likely to happen on lines with words in all caps such as titles that are nearly the full line width. When PRINT encounters such a line, it displays an error message and then prints the line which will extend beyond the right margin. The error message gives the line number in the text file which simplifies the task of locating and altering the line.

A similiar problem can occur with centered text having predominently wide characters such as centered, capitalized titles. PRINT will attempt to adjust the text so that it occupies the same percentage of the printed column width as its character count is a percentage of the number of characters between the left and right margins. If the character widths are such that the text will not fit within this space, it will overflow to the right and result in an off-center title. An error message is not displayed unless the text also overflows the right margin as described above. A solution to this problem is to put two or more blanks between the words of centered titles thus giving PRINT more room in which to fit the title text.

5.9.3 Additional Considerations

When using TYPE=16, which selects the proportional font with inter-character spacing always set to zero, the width of the space character is quite narrow. As a result, centered text will be printed substantially to the left of center when using TYPE=16. The best way to overcome this problem is to use TYPE=PROP for titles and centered text. TYPE=16 will be found to be virtually useless in columnar text.

5.10.1 Combinations of TYPE, JUSTIFY, and PROPITCH Attributes

The following table shows how various combinations of TYPE, JUSTIFY, and PROPPITCH will be interpreted by the PRINT program for the NEC PC-8023 printer. For conbinations which are not intended to occur, the interpretation is listed as "non-functional".

	TYPE	JUSTIFY	PROPPITCH	INTERPRETATION
1.	PICA	NONE	na	Print in pica font with ragged right.
2.	PICA	FLUSH	na	Print in pica font with flush justify.
3.	PICA	PROP	na	Non-functional. Does same as 2.
4.	ELITE	NONE	na	Print in elite font with ragged right.
5.	ELITE	FLUSH	na	Print in elite font with flush justify.
6.	ELITE	PROP	na	Non-functional. Does same as 5.
7.	COND	NONE	na	Print in condensed font with ragged right.
8.	COND	FLUSH	na	Print in condensed font with flush justify.
9.	COND	PROP	na	Non-functional. Does same as 8.
10.	16	NONE	na	Print in proportional font with ragged right.
11.	16	FLUSH	na	Non-functional. Prints incorrectly because proportional characters are not fixed width.
12.	16	PROP	na	Non-functional. Does same as 11.
13.	PROP	NONE	<cpi>	Print in proportional font with ragged right at specified pitch.
14.	PROP	FLUSH	<cpi>	Non-functional. Does same as 15.
15.	PROP	PROP	<cpi>	Print in proportional font with proportional justification at specified pitch.

Notes:
1. 10 may be specified in place of PICA with the same result.
2. 12 may be specified in place of ELITE with the same result.
3. COMP and 17 may be specified in place of COND with the same result.
4. OFF may be specified in place of NONE with the same result.

NEC PC-8023 Character Code and Print Font Chart

Note: The 5 columns are as follows from left to right:
Hexadecimal character code, 10/PICA, 12/ELITE, 17/COMP/COND, and PROP fonts.

20					50	P	P	P	P	80	_	_	_	_	B0	∞	∞	∞	∞	E0	=	=	=	=
21	!	!	!	!	51	Q	Q	Q	Q	81	_	_	_	_	B1	ɜ	ɜ	ɜ	ɜ	E1	⊦	⊦	⊦	⊦
22	"	"	"	"	52	R	R	R	R	82	▬	▬	▬	▬	B2	γ	γ	γ	γ	E2	‡	‡	‡	‡
23	#	#	#	#	53	S	S	S	S	83	▬	▬	▬	▬	B3	4	4	4	4	E3	⫞	⫞	⫞	⫞
24	$	$	$	$	54	T	T	T	T	84	▬	▬	▬	▬	B4	5	5	5	5	E4	◢	◢	◢	◢
25	%	%	%	%	55	U	U	U	U	85	▬	▬	▬	▬	B5	6	6	6	6	E5	◣	◣	◣	◣
26	&	&	&	&	56	V	V	V	V	86	▬	▬	▬	▬	B6	ϵ	ϵ	ϵ	ϵ	E6	◥	◥	◥	◥
27	'	'	'	'	57	W	W	W	W	87	▬	▬	▬	▬	B7	ρ	ρ	ρ	ρ	E7	◤	◤	◤	◤
28	(	(	(	(	58	X	X	X	X	88	|	|	|	|	B8	σ	σ	σ	σ	E8	♠	♠	♠	♠
29	)	)	)	)	59	Y	Y	Y	Y	89	|	|	|	|	B9	ψ	ψ	ψ	ψ	E9	♥	♥	♥	♥
2A	*	*	*	*	5A	Z	Z	Z	Z	8A	|	|	|	|	BA	Ω	Ω	Ω	Ω	EA	♦	♦	♦	♦
2B	+	+	+	+	5B	[	[	[	[	8B	|	|	|	|	BB	Γ	Γ	Γ	Γ	EB	♣	♣	♣	♣
2C	,	,	,	,	5C	\	\	\	\	8C	|	|	|	|	BC	◊	◊	◊	◊	EC	●	●	●	●
2D	-	-	-	-	5D	]	]	]	]	8D	|	|	|	|	BD	δ	δ	δ	δ	ED	○	○	○	○
2E	.	.	.	.	5E	^	^	^	^	8E	|	|	|	|	BE	κ	κ	κ	κ	EE	/	/	/	/
2F	/	/	/	/	5F	_	_	_	_	8F	+	+	+	+	BF	Σ	Σ	Σ	Σ	EF	\	\	\	\
30	0	0	0	0	60	`	`	`	`	90	⊥	⊥	⊥	⊥	C0	α	α	α	α					
31	1	1	1	1	61	a	a	a	a	91	⊤	⊤	⊤	⊤	C1	ν	ν	ν	ν					
32	2	2	2	2	62	b	b	b	b	92	⊣	⊣	⊣	⊣	C2	⊿	⊿	⊿	⊿					
33	3	3	3	3	63	c	c	c	c	93	⊢	⊢	⊢	⊢	C3	β	β	β	β					
34	4	4	4	4	64	d	d	d	d	94	‾	‾	‾	‾	C4	ξ	ξ	ξ	ξ					
35	5	5	5	5	65	e	e	e	e	95	_	_	_	_	C5	η	η	η	η					
36	6	6	6	6	66	f	f	f	f	96	|	|	|	|	C6	θ	θ	θ	θ					
37	7	7	7	7	67	g	g	g	g	97	|	|	|	|	C7	ι	ι	ι	ι					
38	8	8	8	8	68	h	h	h	h	98	⌐	⌐	⌐	⌐	C8	±	±	±	±					
39	9	9	9	9	69	i	i	i	i	99	¬	¬	¬	¬	C9	υ	υ	υ	υ					
3A	:	:	:	:	6A	j	j	j	j	9A	⌊	⌊	⌊	⌊	CA	π	π	π	π					
3B	;	;	;	;	6B	k	k	k	k	9B	⌋	⌋	⌋	⌋	CB	Λ	Λ	Λ	Λ					
3C	<	<	<	<	6C	l	l	l	l	9C	⌐	⌐	⌐	⌐	CC	²	²	²	²					
3D	=	=	=	=	6D	m	m	m	m	9D	⌐	⌐	⌐	⌐	CD	⊕	⊕	⊕	⊕					
3E	>	>	>	>	6E	n	n	n	n	9E	⌊	⌊	⌊	⌊	CE	-	-	-	-					
3F	?	?	?	?	6F	o	o	o	o	9F	⌋	⌋	⌋	⌋	CF	τ	τ	τ	τ					
40	@	@	@	@	70	p	p	p	p	A0					D0									
41	A	A	A	A	71	q	q	q	q	A1	≥	≥	≥	≥	D1	¢	¢	¢	¢					
42	B	B	B	B	72	r	r	r	r	A2	·	·	·	·	D2	ω	ω	ω	ω					
43	C	C	C	C	73	s	s	s	s	A3	⁑	⁑	⁑	⁑	D3	≈	≈	≈	≈					
44	D	D	D	D	74	t	t	t	t	A4	≤	≤	≤	≤	D4	⌟	⌟	⌟	⌟					
45	E	E	E	E	75	u	u	u	u	A5	/	/	/	/	D5	7	7	7	7					
46	F	F	F	F	76	v	v	v	v	A6	·	·	·	·	D6	8	8	8	8					
47	G	G	G	G	77	w	w	w	w	A7	↑	↑	↑	↑	D7	9	9	9	9					
48	H	H	H	H	78	x	x	x	x	A8	½	½	½	½	D8	ι	ι	ι	ι					
49	I	I	I	I	79	y	y	y	y	A9	↓	↓	↓	↓	D9	φ	φ	φ	φ					
4A	J	J	J	J	7A	z	z	z	z	AA	←	←	←	←	DA	≠	≠	≠	≠					
4B	K	K	K	K	7B	{	{	{	{	AB	→	→	→	→	DB	×	×	×	×					
4C	L	L	L	L	7C	¦	¦	¦	¦	AC	+	+	+	+	DC	▪	▪	▪	▪					
4D	M	M	M	M	7D	}	}	}	}	AD	(	(	(	(	DD	ο	ο	ο	ο					
4E	N	N	N	N	7E	~	~	~	~	AE	)	)	)	)	DE	ζ	ζ	ζ	ζ					
4F	O	O	O	O	7F					AF	¼	¼	¼	¼	DF	λ	λ	λ	λ					
																μ	μ	μ	μ					

5.10.3 Display Character Chart for NEC-8023 Version of WPEDIT

```
                WORDPIC Editor  1.0   (C) Copyright 1983 M.T.U.
File= ALLSCRNCODES.W                                  Col=  1  Line=    1  Top=    1

        ALL 256 POSSIBLE CHARACTER CODES AND HOW THEY APPEAR ON THE SCREEN

00 ■     10 ■     20        30 0     40 @     50 P     60 `     70 p
01 ■     11 ■     21 !      31 1     41 A     51 Q     61 a     71 q
02 ■     12 ■     22 "      32 2     42 B     52 R     62 b     72 r
03 ■     13 ■     23 #      33 3     43 C     53 S     63 c     73 s
04 ■     14 ■     24 $      34 4     44 D     54 T     64 d     74 t
05 ■     15 ■     25 %      35 5     45 E     55 U     65 e     75 u
06 ■     16 ■     26 &      36 6     46 F     56 V     66 f     76 v
07 ■     17 ■     27 '      37 7     47 G     57 W     67 g     77 w
08 ←     18 ■     28 (      38 8     48 H     58 X     68 h     78 x
09 ***   19 ■     29 )      39 9     49 I     59 Y     69 i     79 y
0A ■     1A ■     2A *      3A :     4A J     5A Z     6A j     7A z
0B ■     1B ■     2B +      3B ;     4B K     5B [     6B k     7B {
0C ■     1C ■     2C ,      3C <     4C L     5C \     6C l     7C |
0D ***   1D ■     2D -      3D =     4D M     5D ]     6D m     7D }
0E ■     1E ■     2E .      3E >     4E N     5E ^     6E n     7E ^
0F ■     1F ■     2F /      3F ?     4F O     5F _     6F o     7F ■

*** Prohibited from being displayed on the screen.
| JUSTPARA | FIND | MARK | CHANGE |  | TABSTOP | PRTRCODE | CENTER | OTHER |
```

```
                WORDPIC Editor  1.0   (C) Copyright 1983 M.T.U.
File= ALLSCRNCODES.W                                  Col=  1  Line=   22  Top=   22

        ALL 256 POSSIBLE CHARACTER CODES AND HOW THEY APPEAR ON THE SCREEN

80 _     90 ┴     A0        B0 ∞     C0 α     D0 ¢     E0 =     F0 ⌣
81 ▄     91 ┬     A1 ≥     B1 ³     C1 ν     D1 ω     E1 ╞     F1 ⌣
82 ▄     92 ┤     A2 ·     B2 γ     C2 ⊿     D2 ≈     E2 ╪     F2 ⌣
83 ▄     93 ├     A3 ✱     B3 ⁴     C3 β     D3 ⌐     E3 ╡     F3 ⌠
84 ▄     94 ─     A4 ≤     B4 ⁵     C4 ξ     D4 ⁷     E4 ◢     F4 ⌣
85 ▌     95 ─     A5 ⁄     B5 ⁶     C5 η     D5 ⁸     E5 ◣     F5 ⌣
86 ▌     96 │     A6 ·     B6 ∈     C6 θ     D6 ⁹     E6 ◥     F6 ⌣
87 ▌     97 │     A7 ↑     B7 ρ     C7 ¹     D7 ∪     E7 ◤     F7
88 │     98 ┌     A8 ½     B8 σ     C8 ±     D8 Φ     E8 ♠     F8 ↘
89 ▌     99 ┐     A9 ↓     B9 ψ     C9 ∀     D9 ≠     E9 ♥     F9 ↗
8A ▌     9A └     AA ←     BA Ω     CA π     DA ×     EA ♦     FA ¶
8B ▌     9B ┘     AB →     BB Γ     CB ∩     DB ⌐     EB ♣     FB ⊡
8C ▌     9C ┌     AC +     BC ο     CC ²     DC ⁰     EC ●     FC ↧
8D ▌     9D ┐     AD ⁽     BD δ     CD ●     DD ζ     ED ○     FD ⌠
8E ▌     9E └     AE ⁾     BE κ     CE ¯     DE λ     EE ⁄     FE
8F ┼     9F ┘     AF ¼     BF Σ     CF τ     DF μ     EF ╲     FF ***

*** Prohibited from being displayed on the screen.
| JUSTPARA | FIND | MARK | CHANGE |  | TABSTOP | PRTRCODE | CENTER | OTHER |
```

6. MAJOR SYSTEM UTILITIES

This chapter describes a small subset of the many application and utility programs available for the MTU-130 computer. The ones that are described here are chosen for their general usefulness in word processing applications of the MTU-130. Except for the programming languages briefly described in section 6.14, all of these utility programs are supplied standard with the MTU-130 on the CODOS Distribution Disk. Following is a a list of the utility programs covered in this chapter:

SECTION	DESCRIPTION
6.2	PRINT_NF = No frills print
6.3	FILES = List all files on disk
6.4	DIR = List selected files on disk
6.5	DELETE = Erase a specified file
6.6	COPYF = Copy single or multiple files
6.7	FORMAT+COPYF = Backup onto disk with different sidedness
6.8	GGET = Get graphic file to screen
6.9	GSAVE = Save screen as a graphics file
6.10	GETDIM = Display dimensions of a graphics file
6.11	GCONVERT = Convert loadable file to graphics file
6.12	ANNOTATE = Interactive graphic titling program
6.13	GREDIT = Interactive graphic editor
6.14	BASIC and other languages

6.1 RUNNING UTILITY PROGRAMS

All of these utility programs must be run from CODOS; they cannot be run from the WORDPIC shell. To leave the WORDPIC shell and return to CODOS, press the function key labelled EXIT while the WORDPIC file display is showing. When CODOS is ready for a command, it will display:

CODOS>

with a flashing cursor to the right of the >. CODOS does not use a formatted screen display like WORDPIC does so the CODOS> prompt could actually be at the beginning of any screen line.

In general, to run any of the utility programs (except ANNOTATE and GREDIT), you simply type in the utility name (it must be in all caps so be sure to engage the CAPS LOCK key) in some cases followed by one or more arguments, and then a carriage return. The meaning and format of the arguments is described in the individual utility writeups.

After a utility is finished (most of these are very simple, single function programs that do their job quickly), it will return to CODOS which again displays its prompt meaning that it is waiting for another command. When you are finished running utility programs and wish to re-enter the WORDPIC shell, simply enter WORDPIC in response to the CODOS> prompt and the index display will be restored.

Many of these utilities involve the use of both disk drives at once. In order to access the disk in a drive, the drive must be "open". This is accomplished with the OPEN command which has a very simple syntax:

OPEN <drivenumber>

where <driveneumber> is the number of the drive to be opened. Thus the CODOS command:

```
OPEN 1
```

will open drive 1 and make it available for use.

In order to change a diskette in a drive, it should first be "closed", the disk changed, and then it must be opened with the new disk in place. The CLOSE command is just like the OPEN command except for its name. If you accidentally remove a disk from the drive without closing, you may not have caused any harm. However, to be safe, you should reOPEN any other disks which are still in the disk drives.

If at any time CODOS prints the following message in response to your command:

```
CODOS ERROR #1
COMMAND NOT FOUND
```

it is because you either misspelled the program name or that program is not on the disk in drive zero. If it is due to a misspelling, you can enter CTRL/B to recall the command, then correct the spelling and press RETURN to re-enter the command. If it is due to a missing program, use COPYF to copy the needed program from your copy of the WORDPIC Distribution Disk onto the disk you are using in drive 0 (see section 6.6).

6.2 PRINT_NF - "No Frills" Print

The PRINT_NF program performs the same function as the customized PRINT program used in the WORDPIC system except that it implements only those features which are identical on virtually any kind of printer. Features that are not universal (such as underlining, variable type pitch, etc.) are either ignored or are translated into a similar feature which nearly any kind of printer can perform. Following is a list of how PRINT_NF handles these other features:

1. The code characters for backspace, UNDERLINE, BOLD, WIDE, SUBSCRIPT, and SUPERSCRIPT are simply ignored.

2. The TYPE selection on Paragraph code lines is ignored. The printer will print in the typeface and pitch to which it is currently set.

3. Figure code lines are ignored. Nothing will be printed for them, i.e., no space is skipped.

4. The BACKUP command on Paragraph code lines is ignored.

5. The BACKPAGE command on Newpage code lines is ignored.

6. The LPI setting on Newpage code lines are ignored. The printer will print in the lines per inch to which it is currently set.

7. A setting of 1.5 for SPACING on a Paragraph code line will be treated the same as a setting of 1 for spacing. Settings of 2 and 3 will function normally.

8. Selecting proportional justification on a Paragraph code line will be treated as if it were flush justification.

9. The OFFSET command on a Paragraph code line is implemented by printing 1 space for each tenth of an inch specified. The actual offset distance depends on the width of the space character in the current pitch setting of the printer.

10. The distance between top-of-forms will be the PAGELENGTH value truncated to the nearest integral number of whole lines at 6 lines per inch. Thus only multiples of .5 inch should be used for PAGELENGTH, assuming the printer is set for 6 lines per inch.

All other features of PRINT, such as headers and footers, page numbers, and variable text substitution are performed in the same way by PRINT_NF.

PRINT_NF is started from CODOS the same way PRINT is. See section 5.2 for the specific command format. During operation, PRINT_NF performs just like PRINT with function keys available for stopping, starting, and aborting. Remember the effect of the printer's buffer, if it has one.

If you have a printer that is not currently supported by a full PRINT program and therefore need to use PRINT_NF exclusively to print your WORDPIC text files, you can rename PRINT_NF to PRINT and then use it directly from the WORDPIC shell without having to exit to CODOS first. This is accomplished with the CODOS RENAME command which has the following form:

```
RENAME <oldfilename> <newfilename>
```

where <oldfilename> is the name of the file being converted (in this case PRINT_NF) and <newfilename> is its new name (PRINT in this case). If a PRINT for some printer already exists on your disk, you will need to rename it to something else (such as PRINT_NEC) first and then rename PRINT_NF.

6.3 FILES - List All Files on Disk

The FILES command can be used to list ALL of the file names on a diskette, not just those with .W and .G extensions. The syntax of the FILES command is:

FILES [<drivenumber>]

where <drivenumber> is the number of the disk drive for which a files listing is desired. Thus the command:

FILES 1

will list all of the file names on the disk in drive 1. If no drive number is specified, the listing is for drive 0. The file names are listed in 6 columns and are not alphabetized. An example of a typical file listing is shown below:

```
CODOS.Z          COMDPROC.Z       SYSERRMSG.Z      SVCPROC.Z        STARTUP.J
DIR.C            IODRIVER.Z       GRAPHDRIVER.Z    PRINTDRIVER.Z    COPYF.C
EDIT.C           MACASM.C         DISASM.C         BASIC.C          SYSLIBNAM.Z
CIL.Z            CILB.Z           IGL.Z            VGL.Z            FORMAT.C
KILL.C           BACKUP.C         SPRINT.Z         MTU130EQU.A      GETGRAY.A
KEYLAY3.B        ANNOTATE.T       DATAMOVER.T      BUG_IN_TERM.T    SERIAL_PORT.T
BACKUP_GOLD.T    VBFONTENTER.T    LITEPEN_BOBO.T   ANNOTATE.B       VBFONTENTER.B
VBASCII.S        WORDPIC.C        PRINT.C          DMXMONBLURB.W    ANNOTATE_BUG.W
OP68000ERR.W     WORDPICSAVES.Z   DTACKNEWS.W      INQUEUE_1.W      INQUEUE_2.W
HIDDENPORT.W     MUSIC_BUG.W      SHEPARDTONES.W   SHEPARD_ROE.W    UNASM.W
SHEPARD.N        SHEPARD.T        SHEPARD.M        BAUDCHECK.W      WORDSHELL.C
NFPRINT.C        BAUDCHECK.C      MDO.T            MDO.A            FILECRC.T
FILECRC.A        FILECRC.C        MDO.C            CRTRADIATION.W   DISTDISK1_5.W
NEWDISK.C        WSBACKUP.C       AUTOTERM.W       DVORAK_KEYB.W    MACLIB6502.A
GETGRAY.L        GETGRAY.C        GETGRAY.O        TESTASM.A        TESTASM.L
MACLIB6502.O     FANCYCLEAR.T     FANCYCLEAR.A     FANCYCLEAR.L     FANCYCLEAR.C
FANCYCLEAR.W     KEYLAY0.B        GREDIT.B         KEYLAY1.B        KEYLAY2.B
DVORAK_STD.G     DVORAK_MOD.G     IODRIVER_ASK.Z   USING_GRAY.W     GRAYPIX.B
GRAYPIX.G        NEWPROD_MIO.W    INQUEUE_HDSK.W   MACLIB6502.T     ONGRAY.C
OFFGRAY.C        GRAYTEST.T       AUTOTERM.A       AUTOTERM.T       GETGRAY.T
BUG_MACASM.W     MACASM_BUG.W     NEWP_DMXASM.W    NEWP_MACASM.W    INQUEUE_DIG8.W
MOTOROLA_P1.W    MCS_TEMPS.W      MAINKEYB.G       FKEYS.G

CODOS>
```

6.4 DIR - List Selected Files on Disk

The DIR command is similar to the FILES command except that it allows you to limit the list to file names that match a certain "pattern" and it lists more information about each file. The syntax of the DIR command is:

DIR <pattern>

where <pattern> is a file name that may include "wildcard characters". A simple case would be to list additional information about a single specific file or simplify verify that it exists on the disk. In such a case the pattern would be the complete file name with extension (if not .C) and drive number (if not drive 0). Thus the command:

DIR FIGURE17.G:1

would tell all about the file FIGURE17.G on drive 1, if it indeed exists there.

Wildcards allow you to list the names of all files that match a pattern name you specify. The * wildcard character will match any string of one or more characters up to the file name extension. Thus for example, the command:

DIR FIGURE*.G:1

will match FIGURE1.G, FIGURE2.G, ..., FIGURE17.G. In effect it matches all file names that begin with FIGURE. A related form is:

DIR *.G:0

which will match all file names on drive 0 that have a .G extension.

DIR *.*:1

will list extended information about all of the files on drive 1.

Note that a pattern of the form: *BAK.W:0 to list all of the backup files (assuming the user has adopted such a naming convention) on drive 0 will not work as expected because the leading * will match all of the main part any file name thus leaving nothing for the BAK to match. The - wildcard character will match any character string up to and including an _ in the name. Thus if you adopt the convention of naming backup files with _BAK tacked on the end, a pattern such as: -BAK.W will list all of the backup word processing file names. A third available wildcard character is the ? which will match any single character. More about DIR can be found in chapter 4 of the CODOS manual in the MTU-130 User's Manual.

Below is an example of the file listing produced by DIR:

```
DIR G*.*:0

GRAPHDRIVER.Z  :0 -  11/24/82   $0005EF
GSAVE.L        :0 -  *UNDATED*  $0085E2
GSAVE.C        :0 -  *UNDATED*  $00059C
GGET.L         :0 -  07-MAY-83  $009F13
GGET.A         :0 -  30-APR-83  $005273
GETDIM.A       :0 -  27-APR-83  $0014CD
GETDIM.L       :0 -  10-MAY-83  $00338A
GETDIM.C       :0 -  10-MAY-83  $000186
GCONVERT.A     :0 -  07-MAY-83  $002A61
GSAVE.A        :0 -  28-APR-83  $004261
GSAVE.O        :0 -  28-APR-83  $004255
GHEADER.A      :0 -  30-APR-83  $0001D9
GGET.O         :0 -  30-APR-83  $005273
GCONVERT.O     :0 -  09-MAY-83  $0029DA
GGET.C         :0 -  07-MAY-83  $0005C1
GCONVERT.L     :0 -  10-MAY-83  $005832
GCONVERT.C     :0 -  10-MAY-83  $00033E
GETDIM.O       :0 -  21-MAY-83  $00148F
GREDIT.B       :0 -  17-MAY-83  $003812

CODOS>
```

The :0 column indicates that the files are on disk drive 0. The column showing - and L indicates the "locked" status the the file (cannot be written or deleted). The date column shows the creation date of the file. The last column shows the length of the file in hexadecimal bytes.

6.5 COPYF - Copy Single or Multiple Files

For most word processing applications, the BACKUP or NEWDISK and COPY functions of the WORDPIC Shell are sufficient for moving information around from disk to disk. There are two limitations however. First, the WORDPIC shell cannot copy files whose extension is other than .W or .G (such as utility programs which have a .C extension) unless BACKUP is used which not only erases the destination disk, but it copies everything from the source. Second, the shell's COPY function can only copy one file at a time. COPYF is a CODOS utility which overcomes these limitations.

The most useful form of the COPYF command is:

COPYF <pattern>

The drive with the source file is always the drive number specified in the pattern (or drive 0 if a drive is not specified) while the drive onto which the copy is to be written is always the "opposite" drive. Thus the command:

COPYF GSAVE.C:0

will copy the GSAVE program from the disk in drive 0 onto the disk in drive 1. Conversely:

COPYF GSAVE.C:1

will copy the file from drive 1 onto drive 0.

The file name used with COPYF can also contain "wildcard characters" just like with the DIR command (see section 6.4). Thus the command:

COPYF *.G:1

will copy all of the graphic files on drive 1 onto the disk in drive 0. If any of the file names already exist on drive 0, they will not be recopied from drive 1. When wildcards are used, COPYF will display the name of each file it finds that matches the pattern and then report whether it already existed or was copied.

There are several other forms of the COPYF command, refer to Chapter 4 of the CODOS manual in the MTU-130 Users Manual for more information.

6.6 DELETE - Erase a Specified File

The WORDPIC Shell is only capable of deleting the files with .W or .G extensions that appear in its index listing. The CODOS DELETE command can be used to delete any file regardless of its extension. The syntax of the DELETE command is:

DELETE <filename>

where <filename> is the full name of the file to be deleted. Thus the command:

DELETE GSAVE.C:0

will delete the GSAVE utility program from the disk in drive 0. The DELETE command is swift and sure; it does its job in a fraction of a second without further prompting so be sure the file name, extension, and drive number are what you intend before pressing RETURN.

6.7 FORMAT+COPYF - Backup Onto Disk With Different Sidedness

Using the BACKUP function key in the WORDPIC Shell only works if the duplicate disk has the same number of sides as the original disk. The following procedure can be used to backup a single-sided disk onto a double-sided disk or vice versa. Note that when going from double- to single-sided disks that there should be at least 50% free space on the double-sided disk; otherwise, there may not be enough room on the single-sided disk to hold everything.

1. Exit the WORDPIC Shell into CODOS by using the EXIT function key. The CODOS> prompt should appear.

2. The disk being backed up must have the FORMAT and COPYF programs on it and must be in drive 0. If this is not the case, copy the necessary programs onto it using COPYF (see section 6.1 and 6.6).

3. Place the disk to receive the copy in drive 1.

4. Type in: FORMAT and a carriage return.

5. Answer "Y" to the "Are you sure ..." question. It will take about 30 seconds to format a single-sided disk and about a minute for a double-sided disk.

6. Answer the "Test for defects ..." question either way; the considerations are the same as for NEWDISK under the WORDPIC Shell.

7. Answer "Y" the "Copy the system?" question. This is absolutely necessary. If it is answered "N", the disk cannot be used in drive 0 and cannot be upgraded later either without reformatting.

8. When the CODOS> prompt reappears, enter: COPYF and a carraige return. The absence or any arguments for COPYF causes it to copy all of the files on the disk in drive 0 to the disk in drive 1. The name of each file will be displayed as it is copied.

9. When going from a double-sided disk with less than 50% free space to a single-sided disk, you may get a "DISK FULL" error message. If this happens, the file being copied when the message appeared will be incomplete and should be deleted from the duplicate disk (see section 6.6). (Note: this will be the case if you try to backup the double-sided WORDPIC Distribution disk onto a single-sided disk.

6.8 GGET - Get Graphic File to Screen

The GGET Utility program is used to "preview" the appearance of a graphic file on the screen without having to print it on paper. Images either smaller than or larger than the 480 wide by 256 high screen dimensions may be previewed. GGET in conjunction with GSAVE (section 6.9) can also be useful for assembling a larger image from two or more smaller images, up to the size of the screen.

The format of the GGET command is:

GGET <filename> [<horizoffset> [<vertoffset>]]

where <filename> is the name (including .G extension and drive number if not 0) of the graphic file to display, <horizoffset> is an optional horizontal offset, and <vertoffset> is an optional vertical offset.

If neither offset is specified, the graphic file will be retrieved to the upper left corner of the screen. If the file dimensions are smaller than the screen, then that part of the screen not covered by the image will be undisturbed. If the file dimensions are larger, then only the topmost or leftmost portion of the image that will fit on the screen will show.

If unsigned offsets are specified, then the effective location of the "upper left corner" of the screen are redefined. For example, if the command:

```
GGET  FIGURE1.G  240 100
```

is entered, the graphic file FIGURE1.G will be read from drive zero and shown on the screen with the corner of the image starting in the exact center of the screen horizontally and 100 dots down from the top vertically. The remaining part of the screen to the left of center and above Y=155 (255-100) center will be undisturbed. If the image is even smaller than 240 wide by 155 high, the part of the screen to the right and below the image will also be undisturbed. If a vertical offset is not specified, it will be zero.

If negative offsets are specified, then it is possible to view any part of an image that may be wider or higher than the screen. For example, if the image FIGURE7.G is 960 dots wide and 500 dots high, the command:

```
GGET FIGURE7.G  -480
```

could be used to view the upper right quadrant of the image. Conversely,

```
GGET FIGURE7.G  0  -250
```

could be used to view the left bottom quadrant. Any offset value can be used so for example,

```
GGET FIGURE7.G  240 125
```

could be used to view a section the center.

6.9 GSAVE - Save Screen as a Graphic File

The GSAVE utility program is used to save part or all of the current image on the screen as a graphic file suitable for further processing or printing by WORDPIC Print. The syntax of the command is:

```
                                  NL
GSAVE  <filename> [{<XL> <XR> <YB> <YT>}]
                                  GC
```

where <filename> is the name of the graphic file to be created and the other arguments will be explained shortly. If the specified file already exists, you will be asked whether it should be overwritten.

If the command is given with no arguments other than the file name, the entire screen will be saved as a 480 wide by 256 high graphic file. If the letters "NL" (no legends) follow the file name, then only the top 240 dot rows will be saved and the height will be 240. If the letters "GC" follow the file name, them the GRIN cursor is made available to select any two opposing corners of the image rectangle to be saved. When the GRIN cursor appears, use the cursor arrow keys to move it to the desired point and press RETURN to select the point. After two corners have been selected, the file is written.

Alternatively, the image location and size on the screen may be specified directly in terms of X and Y coordinates. Four numbers are needed: the X coordinate of the left edge (XL), X coordinate of the right edge (XR), Y coordinate of the bottom edge (YB), and Y coordinate of the top edge (YT). The elements of the left-right pair or the bottom-top pair may be swapped if desired with no effect.

When GSAVE prompts for information, it first saves that portion of the screen used for prompting. After the prompt and operator response, if any, the screen is restored before the image is saved. The width attribute written into the graphic file header may be any value, not just a multiple of 8. The density attributes written are 72 dots/inch in both directions.

6.10 GETDIM - Display Dimensions of a Graphic File

The GETDIM utility may be used to display the header of a graphic file or to determine if a given file is indeed a graphic file. The syntax of the command is:

GETDIM <filename> [<filename> ...]

where <filename> is the name of a graphic file. Any number of file names may be specified, separated by blanks.

An example of the the format of the display line produced for each file is shown below:

```
FIGURE1.G      Type=$00  Width=370 Height=240  H-Density=72  V-Density=72
```

The type attribute identifies the organization of graphic data in the file with $00 (hexadecimal 00) being a standard left-to-right, top-to-bottom, black-and-white bit image.

6.11 GCONVERT - Convert Loadable File to Graphic File

Prior to the release of WORDPIC 1.0, graphic files were commonly just saved blocks of memory, written either by a CODOS SAVE command or by a DMXMON SAVE command. These files lack a graphic header and thus require the use of HEIGHT= and WIDTH= commands on Figure code lines unless the 480 x 256 default is acceptable. Also, they cannot be processed by GGET. The GCONVERT utility may be used to add a graphic header to such files for enhanced compatibility with WORDPIC.

The syntax of the command is:

GCONVERT <filename>

where <filename> is the name of the file to convert. If the file was created by a CODOS SAVE command, the width will be set to 480 and the height will be determined by the length of the file. If the file was created by a DMXMON SAVE command (Datamover memory save), you will be asked what the width should be. The height is then determined by the length of the file. The horizontal and vertical density fields in the graphic header will always be set to 72 dots per inch. Note that GCONVERT assumes that the file was created as a single block save.

6.12 ANNOTATE - Interactive Graphic Titling Program

ANNOTATE is a BASIC program for adding headlines, captions, simple linework, etc. to a graphic file. It prints letters on the screen with independently selectable horizontal magnification, vertical magnification, horizontal line thickness, and vertical line thickenss. The GRIN cursor allows you to place the characters anywhere on the screen. ANNOTATE is also capable of drawing lines and arrowheads to point out items of interest. The character shape database can be either the STDVBFONTS.S file included on the WORDPIC Distribution Disk or a user defined database created with the VBFONTENTER program, available on User Group Disk #2.

6.12.1 Starting ANNOTATE

The ANNOTATE program should be on a diskette which includes BASIC.C, CIL.Z, IGL.Z, GGET.C, GSAVE.C and STDVBFONTS.S. The first three of these (BASIC.C, CIL.Z, and IGL.Z) are NOT included on the WORDPIC Distribution Disk and thus will have to be copied from your MTU-130 Distribution Disk to your working disk. Note that the version of ANNOTATE supplied with WORDPIC is slightly different from ANNOTATE on User Group disk #2; it has been modified to use graphic files with graphic headers.

Since ANNOTATE is a BASIC program rather than a machine language program, MTU BASIC must first be loaded and in control. This may be accomplished by entering:

```
BASIC
```

in response to a CODOS> prompt. To start ANNOTATE running, enter:

```
RUN "ANNOTATE"
```

when BASIC displays its READY message. Be sure to include the double quotes (") as shown. If ANNOTATE had been run previously and still remains in memory, it may be restarted simply by entering: RUN.

6.12.2 Startup Dialog

After ANNOTATE is started, it will clear the screen, identify itself, and display the following:

```
Do you want to use the standard ASCII fonts file (Y/N)?
```

Type a Y and RETURN if the standard fonts file (STDVBFONTS.S) is suitable. Type an N if you want to use a different font file that has been previously prepared by VBFONTENTER or another program. Assuming Y was entered, ANNOTATE will next ask:

```
Enter the character matrix size desired (1=5x7, 2=7x9, 3=9x11) -?
```

Enter the digit associated with the base character size desired. The standard font file actually contains three complete character fonts. The smallest (1=5x7) is the same size as normal screen characters and small enough to fit 80 characters on a line and 25 lines on the screen. The medium size (2=7x9) font allows up to 60 characters on a line and approximately 20 lines. The largest size (3=9x11) allows up to 48 characters on a line and about 16 lines.

ANNOTATE will next ask:

Enter width factor, height factor, stroke width, stroke height - ?

Each of these 4 entries are integer numbers separated by commas. The first two numbers define the final character size. The base character sizes described earlier have width and height factors of 1. For larger characters, the factors entered will multiply the basic matrix width and height by the specified factor. The factors must be integers. There is no enforced upper limit on character size but the practical limit is 480 for the product of matrix width and width factor and 256 for the product of matrix height and height factor which would fill entire screen for a single character.

The last two numbers define the boldness of the strokes used to draw the characters. They actually specify the stroke width and height in dots so 1,1 would specify the thinnest possible strokes. If the boldness numbers are different, the visual effect is similar to that of a calligraphic pen held in either a horizontal or vertical orientation. Normally you should not specify a stroke width or height much greater than the character width or height factors. Otherwise, the strokes will be so bold that they would overlap and blurr the character shape.

Next ANNOTATE will ask:

Do you want to save the resulting image (Y/N)?

If you want to save the screen contents on disk when you exit ANNOTATE, enter Y. If N is entered, there will be no oppertunity to save the image. Assuming Y was entered, ANNOTATE will next ask:

Enter image file name in double quotes -?

Enter the desired image file name with " on each side. If the file does not already exist, it will be created when you exit ANNOTATE. Be sure to include a .G extension. If the file does exist, it will be overwritten thus allowing ANNOTATE to be used to update an existing image file. ANNOTATE creates image files by calling GSAVE to do the actual saving. ANNOTATE will now ask:

Do you want to load an image file (Y/N)?

If you already have a graphic file image that you wish to annotate, enter Y. If N is entered, you will be starting with a blank screen. Assuming that Y was entered, ANNOTATE will next ask:

Enter image file name in double quotes -?

Enter the desired graphic file name, including .G extension, with " on each side. This must be a graphic file with a header. If the file does not have a graphic header, GCONVERT should be run on it first to add the header (see section 6.11). The screen is cleared before the file is loaded. If the dimensions of the graphic file are smaller than the screen, the file will be loaded into the upper left corner of the screen. In such cases you may specify an offset following the file name to shift the image as it is loaded. For example, if the file name is specified as:

"FIGURE_23.G 200 100"

then it will be shifted 200 dots right and 100 dots down as it is loaded thus providing additional room above and to the left for annotation. See the description of GGET in section 6.8 for a detailed description of the offsets.

6.12.3 Performing the Annotation

Once the preliminaries are complete, you will be presented a cross-hair cursor in the center of the screen. The cross point of the cursor may be moved in any of the four directions by pressing the corresponding cursor key. The movement may be made faster by holding down SHIFT while pressing a cursor key. Pressing any other key will be interpreted as a character to be drawn of as a command.

When a character key is pressed, the entry in the font file corresponding to its code will be drawn. For characters in the standard STDVBFONTS.S file, the lower left corner of the character drawn will be placed at the cursor cross point. After drawing, the horizontal position of the cursor will be advanced into position for drawing another character with proper horizontal spacing. The cursor may be moved at any time so there is no restriction whatsoever on where characters are drawn. Vertical spacing for text will have to be eyeballed.

When the thickness parameters are greater than 1, the lower left endpoint of the character stroke will be centered on the cursor cross point. Thus if the thickness parameters were 3,3, the outside of the character will be one dot below and one dot to the left of the cursor. When a thickness parameter is even, it will be unbalanced above and toward the right.

Errors may be corrected by moving the cursor so that it touches the lower left corner of the error character and then pressing the RUB OUT key. This action clears a rectangle equal to the character cell size times the width and height factors entered previously. After rubbing out, the correct character may be entered. The rub out function may be used for clearing anything on the screen, not just characters drawn with ANNOTATE.

While ANNOTATE is running, 5 function keys are active even though their legends are not shown (to maximize the area available for drawing). Their functions are as follows:

f1 - Save the present screen contents (if specified earlier) and exit to BASIC. You will be asked to point out the upper left and lower right corners of the image area to be saved. Move the cursor to a corner and press RETURN to select it. When both corners have been selected, the image is saved.

f2 - Move the "drawing pen" to the cursor crosspoint without drawing.

f3 - Draw from the previously moved or drawn point to the cursor crosspoint.

f4 - Erase from the previously moved or drawn point to the cursor crosspoint.

f5 - Change the font or the magnification and thickness selected earlier. The portion of the screen covered by the prompts and your answers will be restored after the new font or sizes are selected.

Arrowheads may be drawn by first placing the cursor where the tip should be, holding down CTRL, and then pressing the cursor key that corresponds to the direction the arrowhead should point. This is useful for pointing to portions of a drawing being annotated and for adding dimension lines to mechanical drawings.

Note that to successfully erase a previously drawn diagonal line, the two endpoints must be exactly the same ones that were used to draw the line originally. You may draw or erase a single dot by moving the cursor to the desired position and then pressing f2 followed by f3 to draw or f4 to erase.

6.13 GREDIT - Interactive Graphic Editor

GREDIT is a BASIC program for creating and editing graphic image files. It differs from ANNOTATE in that a "graphic database" is maintained which describes the image in terms of graphic primitives such as lines and symbols rather than as a raw dot matrix. Thus powerful graphic editing functions such as moving entire shapes or titles can be performed. GREDIT also has a "subshape" capability where a set of lines can be given a name and then "called up" as a unit when needed in different areas of the screen.

6.13.1 Starting GREDIT

The GREDIT program should be on a diskette which also includes BASIC.C, CIL.Z, and IGL.Z. These files are NOT included on the WORDPIC Distribution Disk and thus will have to be copied from your MTU-130 Distribution Disk to your working disk. Note that the version of GREDIT supplied with WORDPIC is slightly different from GREDIT on the MTU-130 Demonstration Disk; it has been modified to produce graphic files with graphic headers.

Since GREDIT is a BASIC program rather than a machine language program, MTU BASIC must first be loaded and in control. This may be accomplished by entering:

```
BASIC
```

in response to a CODOS> prompt. To start GREDIT running, enter:

```
RUN "GREDIT"
```

when BASIC displays its READY message. Be sure to include the double quotes (") as shown. If GREDIT had been run previously and still remains in memory, it may be restarted simply by entering: RUN.

6.13.2 General Description

The GREDIT Program is a menu driven application program. It uses the legend boxes as the menus and the function keys as the means of selection. In the next section, descriptions of the various menus and the function of each selection are given.

The function of the GREDIT program is to provide a simple graphics editor. It allows you to create and edit a graphics image. Where a word processor allows you to edit words, the GREDIT program allows you to edit lines in an image.

The image manipulated by the GREDIT program will consist of lines and characters. The lines and characters to be drawn will be specified by you when you enter or edit the image. The program records where the lines and characters are positioned using screen coordinates. This means the image may occupy a coordinate range of 0 to 479 horizontally, and 0 to 255 vertically. Because the GREDIT program uses some of this area for the legends and prompt messages, the usable image area is slightly less. The usable area is about 10 to 479 horizontally, and 30 to 250 vertically.

What is actually created and displayed by this program is a simple graphics data base. This graphics data base consists of a sequence of commands which specify how the image should be drawn. These commands will be referred to as "entries" in the data base.

Three fundamental drawing functions are provided for creating an image. These are move to a point, draw to a point, and draw a character string. An individual line is specified by moving to the starting point, then drawing to the end point. It is also possible to draw a line by drawing from the previous end point to a new end point. A character string is drawn at a desired point by first moving to that point and then drawing the character string. Every time one of these functions is used, an entry describing the operation is added to the data base, and then the operation is actually done on the screen. At any time the screen may be cleared and the entries accumulated in the data base may be re-executed to re-draw the screen.

The GREDIT program has five types of data base entries to support the three drawing functions just described. There are two types of entries which perform the move function. The first is called a MOVE entry. In this entry, the point to move to is specified explicitly. The second is called an RMOVE entry. In this entry, the point to move to is specified relative to the previous end point. There are also two types of entries which perform the draw function. These are called DRAW and RDRAW entries. The draw entries will draw from the previous end point to the end point specified in the entry. In the DRAW entry, the end point is specified explicitly. In the RDRAW entry, the end point is specified relative to the starting point. The fifth entry performs character string drawing. This is called a LABEL entry. It will draw the character string contained in the entry, starting at the previous end point. By using the appropriate combination of entries, almost any simple image can be created.

The GREDIT program also has subshape capabilities. A subshape is a collection of entries which have been given a name. A subshape is most useful when a certain piece of an image appears in several places. You first define how this piece (i.e., subshape) is drawn and assign it a name. Typically subshapes will only use RMOVE, RDRAW, and LABEL entries. You then call for the subshape to be drawn at each of the desired locations. This saves entering the same sequence of entries more than once in the data base.

There are two types of entries which support the subshape capability. The first is the SHAPE entry which is part of the subshape definition. This entry defines the name of the subshape whose entries appear in the data base immediately following the SHAPE entry. The SHAPE entry will contain the 6 character subshape name. The second type of subshape entry is the DRSHP entry. This is the entry in the sequence which "calls up" a subshape by name to be drawn. The DRWSHP entry contains the 6 character subshape name to be drawn. The subshape will be drawn starting at the previous end point.

For greater flexibility, the GREDIT program allows a subshape to call other subshapes (up to 20 levels deep). For example, a subshape called "BOX" may contain DRSHP entries for another subshape called "SQUARE". However, a subshape must not call itself, i.e., the "BOX" subshape must not contain a DRSHP BOX entry.

There is one other type of entry supported, which is the END entry. This entry is found at the end of the entries for each subshape, and at the end of the data base. The END entries are controlled automatically by the GREDIT program.

The GREDIT program organizes the data base into two parts. Starting at the beginning of the data base are all the subshape definitions and their entries. Following the subshape entries come the <u>main drawing entries</u>. This is where the entries which draw your image begin. When you call for the entire image to be drawn, the GREDIT program will start drawing with the first main drawing entry. The subshape entries are placed before the main drawing entries to make the subshapes more easily found by the operator and more quickly found by the program.

For editing purposes, the GREDIT program maintains a current entry pointer. Many of the editing functions involve the current entry in some way. Also, during most of the editing fuctions, a portion of the data base is displayed in the upper right corner of the display. This display will show the two entries preceeding the current entry (if any), followed by the current entry, followed by the two entries after the current entry (if any). The current entry will always be indicated by an arrow.

Entries are added to the data base using the INSERT menu. Entries will be inserted before the current entry. As each entry is inserted, it is also drawn on the display. This allows you to see what you are inserting. Editing functions include DELETE and ADJUST. The DELETE function removes an entry or entire subshape from the data base. An entry is changed by deleting it and then inserting the new entry in its place. The ADJUST function allows you to adjust all appropriate entries by the same amount.

To allow you to reposition the current entry pointer, there is a separate JUMP menu. You may jump to the entry at the beginning or end of the main drawing entries, or any subshape. You can also jump forward or backward in the data base a specified number of entries. You can also search for a point (FIND PNT function). The restriction is that this point must be the beginning point (i.e., the end point of the previous entry) of an entry. If found, the current entry pointer points to the next entry. There is also a DRAW menu to allow you to redraw the entire image or to redraw the image in increments.

There are functions to load and save all or portions of the data base to and from disk. When saving, you may write all of the data base, all of the subshapes, or just selected subshapes to disk. When loading, you may load the entire data base, or just add the main drawing entries in the file on disk to those in the current data base. Finally there are disk functions and some general purpose functions, such as listing the subshape names. A completed image in its bit image form can also be saved as a graphic file for printing as a figure by WORDPIC.

The functions provided should be sufficient to allow you to deal with simple gaphics images. The best way to learn how use GREDIT is through actual use. Refer to the following sections for details on the various functions in each of the menus.

6.13.3 First Main Menu

In this menu the screen will contain a scale across the top and left side of the screen. The small tic marks represent intervals of 5 dots, and the larger marks represent 25 dots. Also, the data base display is shown in the upper right corner of the screen. The following functions are presented in this menu:

LEGEND	FUNCTION
ADVANCE	Advance the current entry pointer to the next entry in the data base.
BACKWARD	Move the current entry pointer backward to the previous entry.
INSERT	Insert new entries between the previous entry and the current entry. See the Insert Menu.
DELETE	Delete the current entry. The next entry will become the current entry. If the current entry is a SHAPE entry, the subshape and all its entries are deleted.

DRAW — Draw all or part of the data base. See Draw Menu

DEF SUB — Adds a new subshape to the data base. You will be prompted for a subshape name which may be up to 6 characters long. A SHAPE entry will be placed in the data base at the end of all the subshape entries already present. An END entry is inserted along with the SHAPE entry to indicate the end of the entries for this subshape. The current entry pointer will point to this END entry so you may immediately use the INSERT command to put entries into this subshape.

JUMP — Move the current entry pointer by various amounts. See JUMP menu.

OTHER — Jumps the Second Main Menu.

6.13.4 Second Main Menu

This menu maintains the same display as in the FIRST MAIN MENU. In this menu, the SAVE, SAVE SUB, and LOAD functions will request a file name. If you wish to include a drive number in this file name, you must enclose the entire file name within double quotes. The following functions are provided in this menu:

LEGEND	FUNCTION
ADJUST	Adjusts MOVE, RMOVE, DRAW, and RDRAW type entries by a fixed amount. You will be asked to identify a point to be moved. Next you will be asked to identify where to move this point. Based on this input, a displacement is calculated. You will then be asked if you wish to adjust the current entry. You may respond by hitting "Y", "N", or "Q" keys to answer "YES", "NO", or "QUIT", respectively. If you answer with "Y", the displacement will be added to the coordinates of the current entry. After this entry is adjusted, succeeding entries will be drawn until the next MOVE, RMOVE, DRAW, or RDRAW entry is found or the end of the data base is encountered. This new entry is made the current entry and the data base entries are redisplayed. You will then be asked if you wish to adjust this new current entry. Responding with "N" performs the same as for "Y" except the entry is left as is. Responding with "Q" returns you to the Second Main Menu.
LIST SUB	Lists the names of all the subshapes defined in the data base.
SAVE	Saves the current graphic data base on disk. You will be asked to enter a file name. If the file specified already exists, you will be prompted again for a file name. If the file does not exist, then all of the current data base is saved in this file. Answering with just a carriage return will return you to the Second Main Menu.
SAVE SUB	Save subshape definitions to a data base disk file. You will first be asked to enter a file name. Next a message about the existence of the file is printed, and you are asked if you wish to save all the subshapes in the current data base. Enter "Y" to save all the subshapes. Note that if the file already exists, then the subshapes will be <u>appended</u> to those in the file. This will overwrite any main drawing entries that may be in the file leaving only subshapes in the file. The program returns to the menu when it is finished writing the subshapes. Enter "N" if you wish to save selected subshapes, or no subshapes. After entering "N", you will be asked to enter a subshape name. Entering a subshape name will cause that subshape definition to be written to the file. The program then prompts you for another subshape name. Answering with just a carriage return will return you to the Second Main Menu.

LOAD — Load a graphic data base file from disk. You will first be prompted to enter an existing file name. Answering with just a carriage return will return you to the Second Main Menu. If the file name entered does not exist, you will be prompted again for a file name. If the file does exist, you will then be asked if you wish to load all of the data base or just the main drawing entries. Answer with "A" to load all the data base. The data base loaded will replace the current one. Answer with "M" to load just the main drawing entries. The entries loaded will be added to the end of the main drawing entries of the current data base.

DISK CMD — Jumps to the Disk Commands Menu.

TO CODOS — Returns to CODOS.

EXIT — Jumps back to the First Main Menu.

6.13.5 Insert Menu

When you enter the Insert Menu, GREDIT determines if you are inserting into the main drawing entries or a SUBSHAPE. If you are inserting into a SUBSHAPE, you will be asked to enter a point from which to draw the SUBSHAPE. At this point the data base is drawn from the beginning of the SUBSHAPE up to but not including the current entry. If you are inserting into the main drawing entries, there will be a pause while a portion of the data base is redrawn. This is done so that the GRIN cursor will appear at the end point of the entry just prior to where entries will be inserted into the data base. Once this is done, the Insert Menu is presented and the GRIN cursor is displayed. You may now insert as many entries as you like while in this menu. Each entry inserted will be drawn as it is entered keeping the image on the screen up to date.

LEGEND	FUNCTION
MOVE	Insert a MOVE entry into the data base. You should first position the GRIN cursor to the point you wish to move to, then press this key.
DRAW	Insert a DRAW entry into the data base. You should first position the GRIN cursor to the point you wish to draw to, then press this key.
RMOVE	Insert an RMOVE entry into the data base. You should first position the GRIN cursor to the point you wish to move to, then press this key.
RDRAW	Insert an RDRAW entry into the data base. You should first position the GRIN cursor to the point you wish to draw to, then press this key.
DRAW SUB	Insert a DRSHP entry into the data base. You will be prompted to enter a subshape name. If the subshape is defined it will be drawn upon insertion of this entry. Responding with just a carriage return will exit back to the Insert Menu without inserting a DRSHP entry. The subshape will be drawn starting at the end point of the previous entry.
LABEL	Insert a LABEL entry into the data base. You will be prompted to enter a label string. This string may be up to 80 characters, which is all that will fit on one line. The string will be drawn starting at the end point of the previous entry.
ADVANCE	Draw the current entry and advance the pointer to the next entry.
EXIT	Jump back to First Main Menu.

6.13.6 Draw Menu

When you first enter the Draw Menu, GREDIT will determine if the current entry pointer is pointing into the main drawing entries or a subshape. If it is a subshape, you will be prompted to enter a point from which to draw the subshape. If it is a main drawing entry, there will be no prompt. In either case, the DRAW MENU then presents the following functions for drawing all or part of the main drawing entries or subshape:

LEGEND	FUNCTION
ALL	Draws all of the subshape or main drawing entries after first clearing the screen.
FROM BEG	Draws from the beginning of the subshape or main drawing entries up to but not including the current entry.
DRAW 10	Draw the next 10 entries. Drawing a subshape counts as one entry.
DRAW 5	Draw the next 5 entries. Drawing a subshape counts as one entry.
DRAW 1	Draw the next entry.
STEP	Draws the current entry and advances the current entry pointer to the next entry.
SAVE IMG	Saves the current screen image as a graphic file suitable for printing. You will be prompted for the name of the graphic file in which the image is to be saved. Just a carriage return will return you to the Draw menu without saving the image. If the file already exists, you will be asked whether it should be overwritten. You will then be asked to select the upper left corner and lower right corner of the image to be saved. To select a corner, move the GRIN cursor to the corner location and press RETURN. After both corners have been selected, that portion of the screen is written to the file and GREDIT returns to the Draw menu.
EXIT	Jumps back to the First Main Menu.

6.13.7 Jump Menu

The Jump Menu presents the following functions to allow you to reposition the current entry pointer in various ways:

LEGEND	FUNCTION
TO BEG	Sets the current entry pointer to the first main drawing entry.
TO END	Sets the current entry pointer to the end of the main drawing entry.
TO SUB	Sets the current entry pointer to the first drawing entry in a subshape.
TO S END	Sets the current entry pointer to the end of the drawing entries in a subshape.

FIND PNT — Finds where a specified beginning point occurs in the data base. You will be asked to enter the point to search for. This point should be the beginning point of one of the entries in the data base. The search will begin with the current entry.

The search is accomplished by drawing (or redrawing) entries one at a time, checking the beginning point of each entry before it is drawn. If a match is found the searching stops. The current entry will be this entry. At this point the data base will be displayed and you will be asked if this is the desired entry. If so, enter Y for "yes". If not, enter N for "no", and searching will resume. You may also enter Q for "quit", if you wish to stop the search at this point. Searching will stop and the query given no matter where the match occurs. The match may be within a subshape.

The search must begin with the current entry pointer pointing in the main drawing entries. It cannot start within a subshape, though it may stop within a subshape. If a match is not found when the end of the data base is reached, the current entry pointer is reset to the beginning of the main drawing entries.

FORWARD — Move the current entry pointer forward a specified number of entries.

BACKWARD — Move the current entry pointer backward a specified number of entries.

EXIT — Jump back to First Main Menu.

6.13.8 Disk Commands Menu

LEGEND	FUNCTION
OPEN 0	Opens drive 0.
OPEN 1	Opens drive 1.
CLOSE 0	Closes drive 0.
CLOSE 1	Closes drive 1.
FILES 0	Clears the screen and executes a "FILES 0" command.
FILES 1	Clears the screen and executes a "FILES 1" command.
DELETE	Deletes a file. You will be prompted to enter a file name to be deleted. If this file name is to include a drive number, it must be enclosed within double quotes.
EXIT	Jumps back to the Second Main Menu.

7.1 WORDPIC SYSTEM CHARACTER CODE CHART

CODE	SHAPE	FUNCTION	CODE	SHAPE	FUNCTION	CODE	SHAPE	FUNCTION
00			30	0		60	`	
01			31	1		61	a	
02			32	2		62	b	
03			33	3		63	c	
04			34	4		64	d	
05			35	5		65	e	
06			36	6		66	f	
07			37	7		67	g	
08	←	Backspace	38	8		68	h	
09		Tab	39	9		69	i	
0A			3A	:		6A	j	
0B			3B	;		6B	k	
0C			3C	<		6C	l	
0D		End-of-line	3D	=		6D	m	
0E			3E	>		6E	n	
0F			3F	?		6F	o	
10			40	@		70	p	
11			41	A		71	q	
12			42	B		72	r	
13			43	C		73	s	
14			44	D		74	t	
15			45	E		75	u	
16			46	F		76	v	
17			47	G		77	w	
18			48	H		78	x	
19			49	I		79	y	
1A			4A	J		7A	z	
1B			4B	K		7B	{	
1C			4C	L		7C	|	
1D			4D	M		7D	}	
1E			4E	N		7E	~	
1F			4F	O		7F		Rubout
20		Space	50	P		80	Printer depen-	
21	!		51	Q		..	dent shape	
22	"		52	R		EF	and function	
23	#		53	S				
24	$		54	T		F0		Under begin
25	%		55	U		F1		Bold begin
26	&		56	V		F2		Wide begin
27	'		57	W		F3		Print pause
28	(		58	X		F4		Under end
29	)		59	Y		F5		Bold end
2A	*		5A	Z		F6		Wide end
2B	+		5B	[		F7	-RESERVED-	
2C	,		5C	\		F8		Subscript
2D	-		5D	]		F9		Superscript
2E	.		5E	^		FA		Paragraph
2F	/		5F	_		FB		Figure
						FC		Newpage
						FD		Tabstop
						FE	-RESERVED-	
						FF	←	Backspace

7.2 GRAPHIC FILE HEADER FORMAT

BYTE #	VALUE	COMMENT
0	$C7	Graphic image file marker (="G"+$80)
1	$00	Image type code ($00=uncompressed bit image)
2	XX	Width in dots, least significant byte
3	XX	Width in dots, most significant byte
4	XX	Height in dots, least significant byte
5	XX	Height in dots, most significant byte
6	XX	Horiz. density in dots per inch, least significant byte
7	XX	Horiz. density in dots per inch, most significant byte
8	XX	Vert. density in dots per inch, least significant byte
9	XX	Vert. density in dots per inch, most significant byte
10-	XX	Image data, 8 dots per byte

For a type code of $00, the image data is organized left-to-right and top-to-bottom with 8 horizontal dots per byte. The leftmost dot on the screen is the most significant bit of the byte. Where the width is not a multiple of 8, the unused bits in the rightmost byte of a row are zero.

7.3 CONTENTS OF THE WORDPIC DISTRIBUTION DISK

FILE NAME	FUNCTION
ANNOTATE.B	The ANNOTATE interactive graphic titling program
BACKUP.C	The BACKUP utility program
CODOS.Z	The CODOS disk operating system kernal
COMDPROC.Z	The CODOS command processor
COPYF.C	The COPYF utility program for 2 or more disrives
COPYF1DRIVE.C	The COPYF utility program for a single disk drive
CPUID.C	Program to display the computer's serial numbers
DIR.C	The DIR utility program
EDIT.C	Standard line-oriented text editor
FORMAT.C	The FORMAT utility program
GCONVERT.C	The GCONVERT utility program
GETDIM.C	The GETDIM utility program
GGET.C	The GGET utility program
GRAPHDRIVER.Z	The CODOS graphic display processor
GREDIT.B	The GREDIT interactive graphic editor program
GSAVE.C	The GSAVE utility program
IODRIVER.Z	The CODOS keyboard and character display processor
KILL.C	The KILL utility program
NEWDISK.C	WORDPIC utility for creating a new disk
PRINTDRIVER.Z	CODOS printer driver for parallel printers
PAGINATE.C	WORDPIC page formatter program
PRINT.A	WORDPIC PRINT program source code (.READ statements)
PRINT.C	WORDPIC PRINT program, PRINTER DEPENDENT
PRINT1.A	WORDPIC PRINT program source code, printer independent portion
PRINT2_xxx.A	WORDPIC PRINT program source code, PRINTER DEPENDENT
PRINT2_NF.A	WORDPIC PRINT program source code, no frills
PRINT_NF.C	The PRINT_NF utility program
SAMPLE1.W	Sample text file for WORDPIC Tutorial Manual
SAMPLE1_FIG.G	Figure file for SAMPLE1.W
STARTUP.J	List of commands to be executed at power-up
STDVBFONTS.S	Standard character fonts for ANNOTATE
SVCPROC.Z	The CODOS superviosr call processor
SYSERRMSG.Z	CODOS error messages
TEMPLATES.W	Templates file for SAMPLE1.W
WORDPIC.C	The WORDPIC Shell
WORDPICSAVES.Z	Save buffers file for WPEDIT
WPEDIT.C	WORDPIC editor program, PRINTER DEPENDENT
WSBACKUP.C	WORDPIC utility for backing up a disk

NOTE: Files listed as being PRINTER DEPENDENT will vary according to the brand of printer the distribution disk is configured for. The default printer support is for the NEC PC-8023A.

GRAPHIC DENSITY CHART

The density attribute on a Figure code line consists of two digits. The first digit specifies the horizontal density in dots per inch and the second digit specifies the vertical density according to the table below:

Digit value	1	2	3	4	5	6	7	8	9
1st, horiz.	60	72	80	96	120	144	160	192	240
2nd, vert.	48	72	80	96	108	144	192	216	288

Any given printer can only print a small subset of the 81 possible combinations of horizontal and vertical density. The chart below lists the densities available on printers currently supported by a PRINT module for WORDPIC.

DOTS / INCH VERT	HORIZ	PRINTERS	DOTS / INCH VERT	HORIZ	PRINTERS	DOTS / INCH VERT	HORIZ	PRINTERS
48	60		108	60		288	60	SANTEC
48	72		108	72		288	72	
48	80		108	80		288	80	SANTEC
48	96		108	96		288	96	SANTEC
48	120		108	120		288	120	SANTEC
48	144		108	144		288	144	
48	160		108	160		288	160	SANTEC
48	192		108	192		288	192	SANTEC
48	240		108	240		288	240	SANTEC
72	60	SANTEC	144	60	SANTEC			
72	72	SANTEC	144	72				
72	80	SANTEC, NEC	144	80	SANTEC, NEC			
72	96	SANTEC, NEC	144	96	SANTEC, NEC			
72	120	SANTEC	144	120	SANTEC			
72	144	NEC	144	144	NEC			
72	160	SANTEC,NEC	144	160	SANTEC,NEC			
72	192	SANTEC	144	192	SANTEC			
72	240	SANTEC	144	240	SANTEC			
80	60		192	60				
80	72		192	72				
80	80		192	80				
80	96		192	96				
80	120		192	120				
80	144		192	144				
80	160		192	160				
80	192		192	192				
80	240		192	240				
96	60		216	60				
96	72		216	72				
96	80		216	80				
96	96	SANTEC	216	96				
96	120		216	120				
96	144		216	144				
96	160		216	160				
96	192		216	192				
96	240		216	240				

INDEX

NOTE: Words which are in all capitals are the names of programs, commands, or function keys. In the case of function keys, the name of the program which uses it will be in parentheses. Other words will have only the first letter capitalized.

@+ 3.12.17, 4.6.1, 4.6.2, 4.6.5
@variables 3.2.18, 5.8, 4.6.5, 4.11

ABORT function key (PRINT) 5.3.4
ACCEPT function key (PAGINATE) 4.5
ACTION function key (WPEDIT) 3.14
ANNOTATE utility program 6.12

BACK PG function key (PAGINATE) 4.8
BACKPAGE= command 3.12.23, 5.6.2, 5.7.8
BACKUP function key (WORDPIC) 2.12
BACKUP= command 3.12.22, 5.7.9
BASIC language 6.12, 6.13
BREAK key 3.2.3, 3.4

CENTER function key (WPEDIT) 3.16
CHANGE function key (WPEDIT) 3.5
CHG DATE function key (WORDPIC) 2.8
CHG DSKO function key (WORDPIC) 2.10
CHG WDTH function key (WORDPIC) 2.8
Code characters 3.6
Code lines 3.7
Command status area 3.2, 3.2.1, 4.5
Commands, on code line 3.12
COPY function key (WORDPIC) 2.7
(WPEDIT) 3.14
COPYF utility program 6.5
CSA see Command status area
CTRL/B 3.2.4
Cursor (WPEDIT) 3.3.1
(WORDPIC) 2.2

DATE? function key (WORDPIC) 2.8
DELETE characters 3.3.3
utility program 6.6
function key (WORDPIC) 2.12
(WPEDIT) 3.14
keyboard key 3.3.3, 3.3.4
lines 3.3.4
DENSITY= 3.12.19, 3.12.21, 5.7.12
DIR utility program 6.4

EDIT function key (WORDPIC) 2.3
Error messages (WPEDIT) 3.2.5
(PRINT) 5.3.5
EXIT function key (WORDPIC) 2.13

FIGURE code line 3.11, 5.6.3
function key (WPEDIT) 3.11
FILE ATT function key (WORDPIC) 2.8
FILE= command 3.12.16, 5.8
FILES utility program 6.3
FIND function key (WPEDIT) 3.4, 3.14
FIND PG function key (PAGINATE) 4.7
FOOT= command 3.12.15
FORMAT utility program 6.7

GCONVERT program 6.11
GETDIM program 6.10
GGET program 6.8
Graphics 3.11, 3.12.16, 3.12.19-21 5.6.3, 5.7.12-14 6.8-13
GREDIT program 6.13
GRIN cursor 6.9, 6.12.3, 6.13.5
GSAVE program 6.9

HEAD= command 3.12.14
Header, graphic file 5.6.3, 5.7.12-14 6.11
HEIGHT= command 3.12.20, 5.7.14, 5.8

INDENT= command 3.12.4, 5.7.15
INSERT characters 3.3.3
keyboard key 3.2.1, 3.3.3, 3.3.4
line 3.3.4

JUSTIFY= command 3.12.10, 3.12.11, 5.6.1, 5.7.1, 5.7.2, 5.8
JUSTPARA function key (WPEDIT) 3.8, 3.13

Keyboard 3.2.3
Keyword 3.12

Landscape mode editing 2.4, 3.2.2
LEFT= command 3.12.1, 5.7.1, 5.8, 5.9.1
LINE function key (WPEDIT) 3.14
LINES= command 3.12.13, 4.3, 4.6.8, 5.6.2, 5.7.5
LN WDTH? function key (WORDPIC) 2.8
LPI= command 3.12.6-8, 3.12.22, 5.7.6, 5.7.9, 5.7.11, 5.8

MACRO function key (WPEDIT) 3.17
MARK function key (WPEDIT) 3.14
MARKMORE function key (WPEDIT) 3.14
Messages, error (WPEDIT) 3.2.5
(PRINT) 5.3.5
MOVE function key (WPEDIT) 3.14

NEW DISK function key (WORDPIC) 2.11
NEWFILE function key (WORDPIC) 2.4
NEWPAGE code line 3.10
function key (PAGINATE) 4.6
(WPEDIT) 3.10
NEXT PG function key (PAGINATE) 4.7
Normal mode editing 2.4, 3.2.2

ODDFOOT= commnad 3.12.15, 4.9, 4.11
ODDHEAD= command 3.12.14, 4.9, 4.11
ODDOFFSET= command 3.12.3, 5.7.1, 5.8, 4.9, 4.11
OFFSET= command 3.12.3, 5.7.1, 5.8, 4.3, 4.11

PAGE function key (WPEDIT) 3.14
PAGELENGTH= command 3.12.12, 5.6.2, 5.7.7, 4.3
PAGINATE program 1.6, 4.
        function key (WORDPIC) 2.5
        startup 4.4
PARA function key (WPEDIT) 3.14
Paragraph code line 3.9, 3.13, 5.6.1
PF1 keyboard key 3.3.5
PRINT function key (WORDPIC) 2.6
      program 1.7, 5.
      startup 5.2
PRINT_NF program 6.2
PRIOR_PG function key (PAGINATE) 4.7
PROPPITCH= command 3.12.9, 3.12.10, 5.7.4, 5.8, 5.9, 5.9.1
PSPACING= command 3.12.8, 5.6.1, 5.7.11, 5.8

QUIT function key (PAGINATE) 4.2, 4.5, 4.11
                  (WPEDIT) 3.19
RECALL function key (WPEDIT) 3.15
REJECT function key (PAGINATE) 4.5, 4.11
REM= command 3.12.24
RENAME function key (WORDPIC) 2.8
REVIND= (see REVINDENT=)
REVINDENT= command 3.12.5, 5.7.16, 5.8, 5.9.1
RIGHT= command 3.12.2, 5.7.1, 5.8, 5.9.1
RUBOUT keyboard key 3.3.3

SAVE function key (WPEDIT) 3.14
SEL DSKO, SEL DSK1 function key (WORDPIC) 2.9
SET VAR function key (PAGINATE 4.6.3, 4.11
SHO VAR function key (PAGINATE) 4.7
SIZE? function key (WORDPIC) 2.8
SPACING= command 3.12.6, 3.12.8, 3.12.22, 5.6.1, 5.7.9-11, 5.8
Special codes 3.3.5
SPLIT function key (WPEDIT) 3.14
SUSPEND function key (PRINT) 5.3.4

TABSTOP code line 3.8, 5.6.4
Templates 4.6, 4.7
TEMPLATES function key (PAGINATE) 4.7
Text window 3.2.2, 4.5
TYPE= command 3.12.9, 5.7.1, 5.7.3, 5.8, 5.9

UPDATE function key (WPEDIT) 3.18
Utility programs 1.8, 6.

Wide mode editing 2.4, 3.2.2
WIDTH= command 2.12.21, 5.7.13
WORDPIC program 1.4, 2.
        Shell 2.
        startup 2.1
WORDPICSAVES.Z file 1.5, 3.1.1
WPEDIT program 1.5, 3.